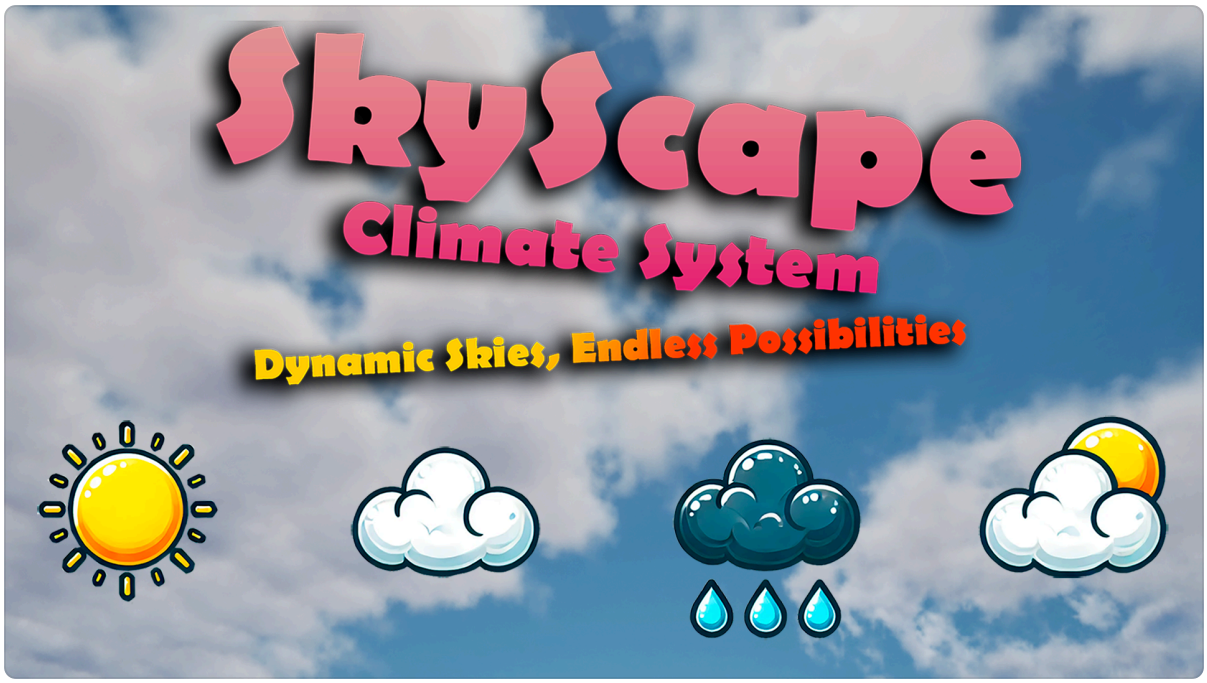
Introduction & FREE Showcase

Download SkyScape:

Download FREE Showcase:Download Here

SkyScape is a **comprehensive solution** that handles many essential **climate and time-management elements** of game development, letting you focus on **core gameplay**. Control **random or fixed weather cycles**, **sync with real time**, and easily configure **custom seasons**.



FREE Showcase!

You can now download a **free showcase** of all **SkyScape's features** from the link below. In this demo, you can experiment with **random or sequential weather cycles**, choose a **default season style**, and decide whether to **sync time with the real world**. Everything shown here is **fully included** in **SkyScape** once you purchase it!

Download Here

A complete weather system with the following features:

Real-Time or Custom Start

- Begin the simulation synced to the actual time in the real world, or define any custom starting hour.

Random or Sequential Cycles

- Automatically cycle through various weather conditions at random intervals or follow a pre-defined sequence.

Fully Customizable Seasons

- Configure a realistic, four-season calendar—or invent fictional seasons with unique month and day lengths, complete with snowfall and rain tied to temperatures.

Flexible Weather Controls

- Switch weathers instantly via an integrated interface, either by the developer or in-game by the player. Set a default style to maintain consistent conditions throughout your levels if desired.

Time and Date Tracking

- Displays and updates the current hour, day, month, and year. These variables can be saved and loaded using customizable slots.

Save System with Toggle

- Choose which parameters (e.g., current weather, date, time) get saved or ignore them altogether. A dedicated button lets you disable saving completely, ideal for testing or simplified setups.

Slot-Based Progression

- Store different pieces of data in separate save slots (e.g., one slot for weather data, another for date/time), providing fine-grained control over what you want to keep persistent.

Quick Integration

- Simply drag the system's blueprint into your level, and you're good to go. No complex setup required.

Whether you need quick real-time synchronization or detailed seasonal transitions, this Weather & Time System provides a powerful, user-friendly toolkit to enhance immersion and flexibility in your projects.

Network Replicated: Yes

- **Windows: Yes**
- **Linux: Yes**
- **Mac: Yes**

Updates and Roadmap

SkyScape v1.0.0 Release Notes(15/01/2025)

SkyScape is here, redefining how you manage dynamic weather and time systems in Unreal Engine 5! Packed with advanced features, customizable options, and seamless integration, SkyScape offers unparalleled control over your game environments.

Core Features

- **Dynamic Weather Systems:** Experience smooth transitions between weather types, with options for sequential or random changes.
- **Customizable Seasons:** Modify durations, weather patterns, and add new fictional seasons. Dynamically switch between seasons in real-time to create immersive worlds.
- **Time Synchronization:** Set the system to sync with real-world time or use custom parameters for full control.
- **Advanced Fog and Lighting:** Includes volumetric fog, directional inscattering, and fully adjustable lighting for enhanced atmosphere.
- **Save System Integration:** Save and load weather states, time, and configurations with slot-based saves.
- **Blueprint Support:** Extensive Blueprint functions and macros for easy customization and real-time updates.

SkyScape v1.1.0 Release Notes(31/01/2025)

SkyScape continues to push the boundaries of dynamic weather and time simulation in Unreal Engine 5 with the release of version 1.1.0! This update introduces new features, improved performance, and greater customization options for developers.

New Features & Enhancements

- **SkyLight Integration:** Full support for SkyLight, enhancing global illumination and lighting realism.
- **Post Process Volume Integration:** Direct control over Post Process Volumes, allowing adjustments to exposure, color grading, and other advanced visual effects.
- **New Cloud Material:** A newly designed cloud material brings even more realism to your skies, providing improved detail and volumetric accuracy.
- **Option to Disable Night Sky:** A new setting allows users to toggle the night sky on or off, providing greater flexibility in different environments.
- **Performance Optimization:** The entire system has been further optimized to improve efficiency, resulting in better performance in large and complex projects.

With these improvements, SkyScape continues to offer one of the most comprehensive and customizable weather systems available for Unreal Engine 5.

SkyScape v1.1.1 Release Notes(31/01/2025)

SkyScape continues to evolve with version 1.1.1, focusing on improving stability and fixing a critical issue that affected developers' experience in Unreal Engine 5.

Bug Fixes

- **Crash Fix (Rain System):** Fixed a bug that caused a crash when dynamically enabling or modifying rain under certain weather configurations. The rain effects are now managed more efficiently to prevent engine crashes.

This update ensures that SkyScape remains a robust and reliable dynamic weather and time system for Unreal Engine 5.

SkyScape v1.2.0 Release Notes(03/02/2025)

SkyScape continues to push the boundaries of dynamic weather and time simulation in Unreal Engine 5 with version 1.2.0. This release introduces exciting new features and enhancements that cater to both single-player and multiplayer projects, ensuring a more immersive and efficient development experience.

New Features & Enhancements

- **Multiplayer Support**:
  Expand your game's horizons with our multiplayer support. SkyScape now offers synchronized weather and time across servers, paving the way for immersive multiplayer environments where every player experiences the same dynamic conditions.
- **Performance Optimizations**:
  Benefit from continued improvements that make SkyScape even more efficient. These optimizations ensure robust performance, even in large-scale and complex projects, so your game's dynamic environment runs smoothly.
- **Realistic Raindrop Effects**:
  Enhance the realism of rainy weather with highly detailed raindrop effects on surfaces. This new feature adds an extra layer of immersion, making your environments feel more natural and engaging during precipitation.
- **User Interface Improvements**:
  A redesigned, more intuitive user interface streamlines configuration and control of SkyScape systems. Enjoy a smoother workflow and easier navigation as you fine-tune weather and time settings.
- **Advanced Save System**:
  Our upgraded save system now supports multiple save slots and sub-slots, allowing for better organization in games with multiple maps or profiles. Each sub-slot stores independent weather states, time, and configurations, enabling dynamic and flexible gameplay experiences.
- **Comprehensive Code Comments**:
  To facilitate customization and debugging, extensive code comments have been integrated throughout the codebase, providing clear insights into SkyScape's functionality.
- **Dynamic Time Initialization**:
  The system now sets the time in the constructor based on the current time, ensuring that your project starts with accurate

and synchronized temporal settings.

- **Automatic Snow Start:**
  Kick off your projects with a wintry ambiance as SkyScape now automatically initiates with snowfall, perfect for creating seasonal atmospheres right from the start.

With these powerful updates, SkyScape v1.2.0 empowers developers to create more immersive, dynamic, and realistic game environments. Enjoy the new features and improvements as you continue to bring your creative visions to life in Unreal Engine 5.

SkyScape v1.3.0 Release Notes(22/02/2025)

SkyScape continues to push the boundaries of dynamic weather and time simulation in Unreal Engine 5 with version 1.3.0. This release introduces a suite of powerful new features and improvements designed to enhance both the developer experience and the immersive quality of your projects.

New Features & Enhancements

- **Weather Presets:**
  Quickly switch between diverse atmospheric conditions with our new, ready-to-use weather presets.
- **Rain and Wind Audio:**
  Elevate the realism of your environments with added sound effects for rain and wind, deepening the immersive experience.

- **Error Indicators:**
  New visual alerts help prevent common configuration mistakes—such as leaving the global pause activated when trying to change weather via functions—ensuring smoother operation and debugging.
- **Enhanced Snow and Rain Effects:**
  Enjoy improved visual fidelity with upgraded snow and rain effects, bringing more natural and engaging weather dynamics to your scenes.

- **Real-Time Editor Preview:**
  Experience instant feedback as rain now updates in the editor without needing to enter play mode. Additionally, activating "start with snow" will display snow directly in the editor.

With these exciting updates, SkyScape v1.3.0 continues to empower developers to create more dynamic, realistic, and immersive game environments in Unreal Engine 5. Enjoy the enhancements as you bring your creative visions to life.

SkyScape v1.4.0 Release Notes(01/03/2025)

We're excited to announce SkyScape v1.4.0 – a major update for Unreal Engine 5 that delivers enhanced realism and streamlines your workflow. Whether you're already using SkyScape or considering it for your next project, this update is packed with innovative features designed to elevate your virtual environments.

New Features & Enhancements

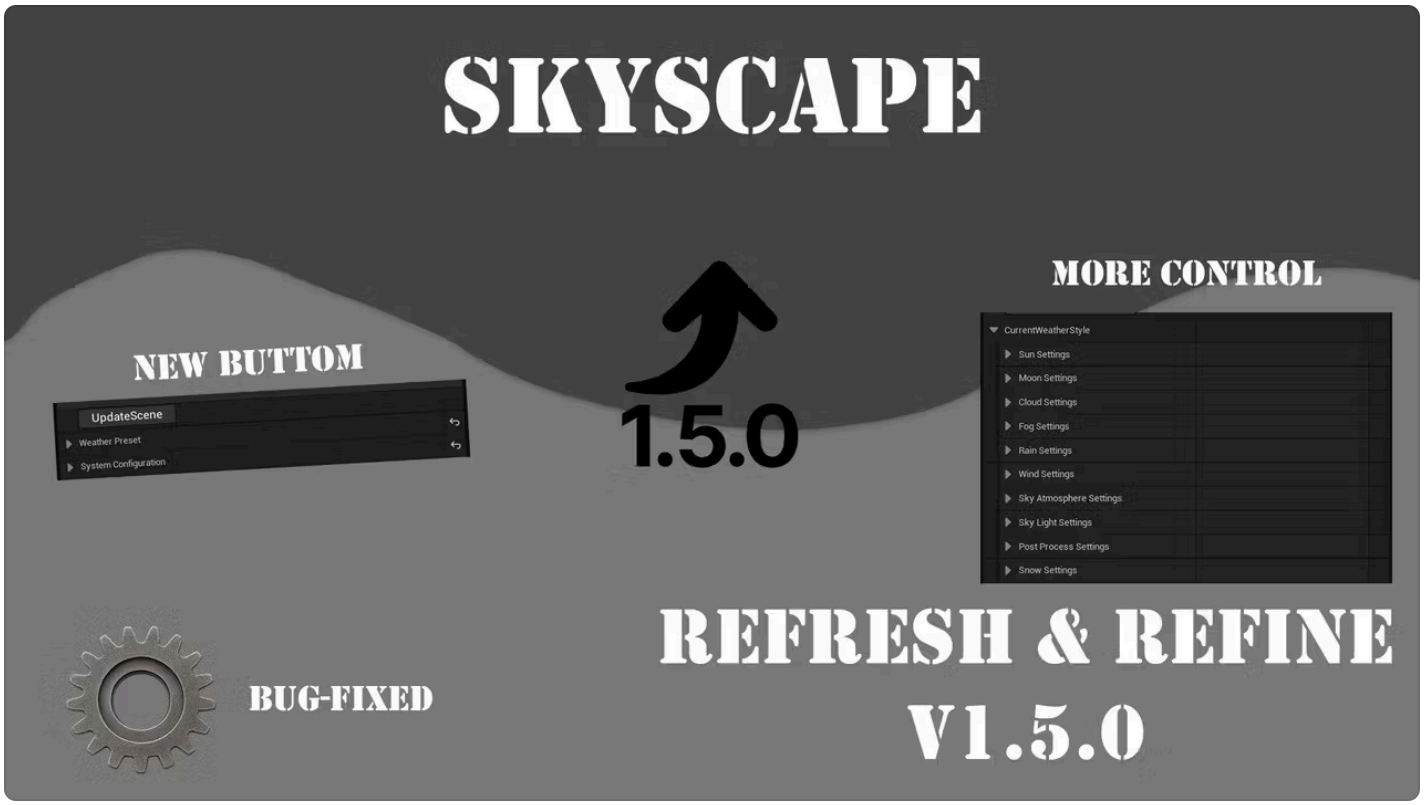
- **Comprehensive Moon System:**
  Experience a fully integrated lunar system that simulates realistic moon phases and dynamic positioning, adding depth and authenticity to your night scenes.
- **Full Star Customization System:**
  Take control of your night sky with advanced star customization tools. Adjust brightness, density, and spatial distribution to create a celestial canvas that perfectly matches your creative vision.
- **Dynamic Eclipse Event:**
  Witness nature's dramatic spectacle as an eclipse is triggered when the sun is obscured by the moon, creating a stunning visual phenomenon that enhances atmosphere and storytelling.
- **Default Rain and Wind Sounds:**
  Enjoy a hassle-free setup with built-in high-quality rain and wind sound effects configured by default, ensuring an immersive auditory experience right from the start.
- **Enhanced Stability and Bug Fixes:**

- Resolved a critical packaging issue that caused crashes when activating rain sounds.
- Fixed a bug where pausing in the Unreal Editor resulted in unexpected crashes, ensuring smoother performance during development.

Upgrade to SkyScape v1.4.0 now to transform your virtual worlds with state-of-the-art weather dynamics and celestial realism. Experience the future of environmental simulation today!

SkyScape v1.5.0 Release Notes (11/03/2025)

We're excited to announce SkyScape v1.5.0 – a major update for Unreal Engine 5 that brings new functionality and improved stability, giving you even greater control over your virtual environments.



New Features & Enhancements

- **Refresh Plugin Button**:
  A new button has been added to refresh the plugin's status, ensuring that your environment reflects changes in real time.
- **Additional Sun Options**:
  Missing sun options have been incorporated to enhance the realism and flexibility of your daylight scenes.
- **Additional Cloud Options**:
  Expanded cloud configuration options provide more control over the appearance and behavior of your sky visuals.
- **Additional Fog Options**:
  New fog settings allow for fine-tuning of atmospheric effects, creating richer and more immersive environments.
- **Additional SkyLight Options**:
  Enhanced SkyLight options improve ambient lighting control for a more balanced and dynamic scene.
- **Additional SkyAtmosphere Options**:
  Additional SkyAtmosphere parameters have been added to further refine the overall sky aesthetics.

- **Crash Bug Fixes**:
  Resolved multiple bugs that were causing unexpected crashes, leading to a more stable and reliable experience.
- **Day Duration Bug Fix**:
  Fixed an issue affecting the day duration timing to ensure a consistent and accurate day-night cycle.
- **Niagara Crash Fix**:
  Addressed a critical bug that caused project crashes due to Niagara when launching the plugin, streamlining your workflow.

Upgrade to SkyScape v1.5.0 now and transform your virtual worlds with enhanced control and stability!

SkyScape v1.6.0 Release Notes (17/03/2025)

We're excited to announce SkyScape v1.6.0 – bringing further refinements and performance improvements to Unreal Engine 5, enhancing your virtual skies with greater flexibility and efficiency.

New Features & Enhancements:

- **Cloud Type Selector**: The boolean option `IsDefaultCloudMaterial` has been replaced with a new selector named `Cloud Type`, offering intuitive control over cloud configurations.
- **Expanded Cloud Materials**: Added 3 new cloud materials and renamed existing ones for clarity:
  - Default
  - SkyScapeMaterial
  - DefaultCartoon
  - SkyScapeMaterialCartoon
  - RealisticClouds
- **Shipping Build Debug Suppression**: Debug messages are now automatically suppressed in shipping builds, ensuring cleaner and more professional deployments.
- **Optimized Sun Rotation Calculations**: Improved efficiency in sun rotation calculations, reducing performance overhead.

- **Improved MarkRenderStateDirty Calls:** Optimized `MarkRenderStateDirty` calls, limiting them to a maximum of once per frame during interpolation for better performance.
- **Updated Cloud Presets:** All presets have been updated for compatibility with the new cloud materials.

Upgrade to SkyScape v1.6.0 and experience optimized performance and greater visual control over your environments!

SkyScape v1.7.0 Release Notes (20/03/2025)

We're excited to announce SkyScape v1.7.0 – taking dynamic weather and time simulation in Unreal Engine 5 to a new level with innovative improvements and fresh features.

New Features & Enhancements:

- **Sequencer Support:** Integrated support for the sequencer, enabling you to create sequences directly with SkyScape.
- **Cloud Materials Bug Fix:** Resolved an issue that prevented cloud materials from hiding correctly.
- **Advanced Sun Rotation System:** Introducing a completely new sun rotation system that calculates sun movement based on the observer's latitude, delivering realistic solar behavior.
- **Sun System Toggle:** Option to switch between the traditional sun system and the new advanced system for greater flexibility.

SkyScape v1.7.1 Release Notes (24/03/2025)

We're excited to announce SkyScape v1.7.1 – a maintenance update that resolves several issues and improves functionality:

- **Corrected Sun Rotation:** The sun rotation has been corrected for both default mode and latitude-based mode, ensuring consistent cardinal axes.
- **Mobility Parameter Added:** A mobility parameter has been added to each component.
- **Clouds and Fog Bug Fix:** Fixed an issue where starting without clouds or fog sometimes resulted in their unexpected appearance.

- **Streamlined Latitude Options:** Latitude options have been restructured into a more compact format to improve usability and interface clarity.
- **New Sun Angle Parameter:** A parameter has been added to adjust the sun angle before nightfall.
- **Sun Offset Variable:** Introduced a variable to control the sun offset relative to real days.
- **Crash Fix:** Resolved a crash that occurred when the start month was set to a value higher than the allowed maximum.
- **Enhanced Constructor Calculations:** Adjustments in the constructor now yield more realistic sun calculations during setup.

Upgrade to SkyScape v1.7.1 and elevate the realism and control of your virtual environments!

SkyScape v1.8.0 Release Notes (06/04/2025)

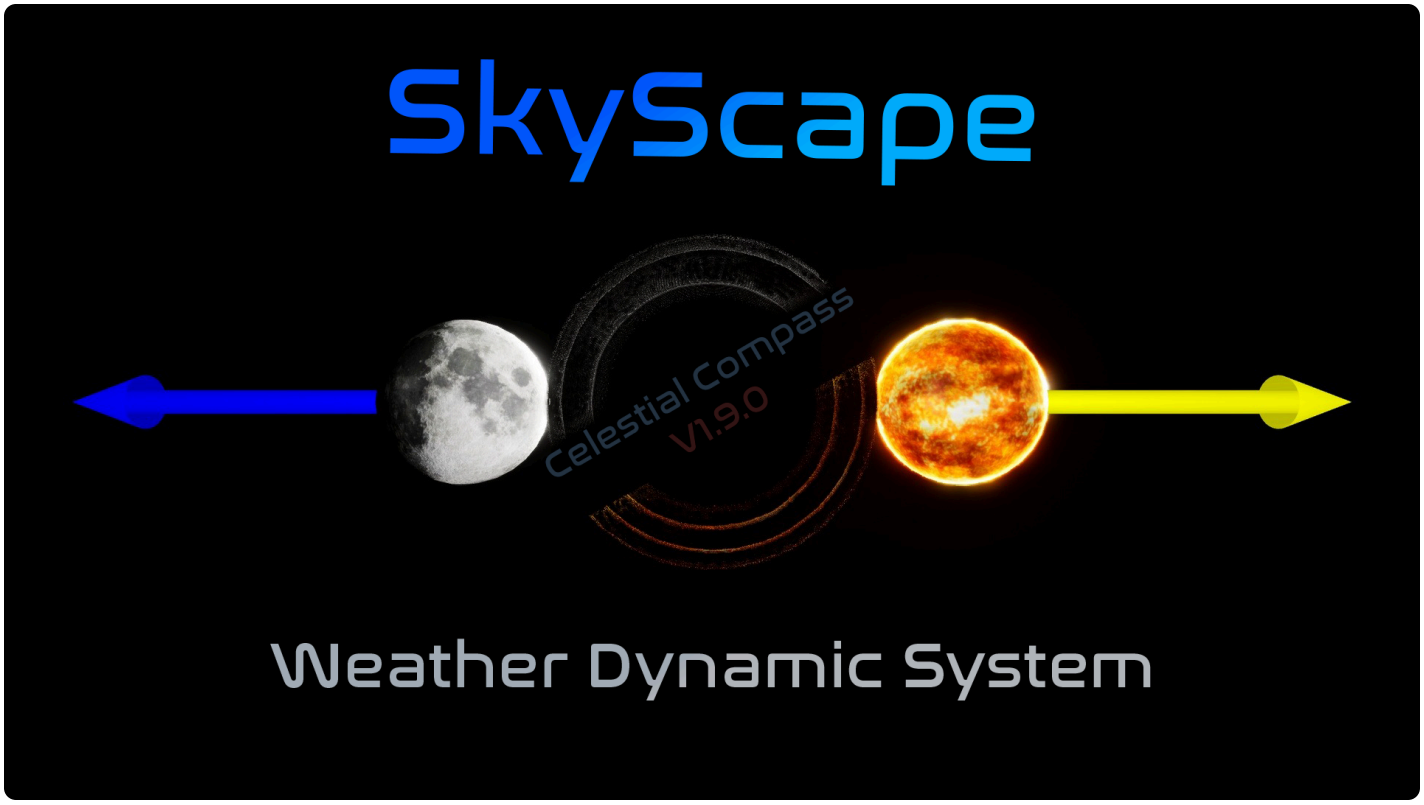
We're thrilled to introduce our latest update, enhancing dynamic environmental interactions and providing more flexible controls for an immersive experience.

New Features & Enhancements:

- **Dynamic Skysphere:** The skysphere now moves towards the player when they move too far away.
- **Rain Blocking Volume:** A dedicated volume has been added to block rain.
- **Preset Toggle:** When presets are active, a button to disable them replaces the "edit current style" option.
- **Sun Options Enum:** A new enumeration allows selection between stationary, basic movement, and advanced movement.
- **Adaptive UI:** Options will be hidden when the sun is set to stationary.

Upgrade to SkyScape v1.8.0 now and experience these exciting improvements in your virtual environments!

Update 1.9.0 Release Notes (24/07/2025): Celestial Compass



SkyScape 1.9.0 introduces significant improvements to the weather system, with a focus on developer usability and more dynamic weather transitions. Notably, this update adds a new **visual debug aid** (a miniature sun-moon orbit display) to help orient developers, and overhauls the weather transition mechanics for greater flexibility. Below is the detailed changelog, organized by category for clarity:

🌟 New Visual Debug Aids (Editor Only)

- **Mini Solar System Visualizer:** When you place the SkyScape actor in a level, it now displays a small **black hole** orb that animates based on time progression. A **mini sun** and **mini moon** orbit around this orb, each with an arrow pointing toward the real Sun and Moon. The mini sun's brightness and angle respond in real-time to the directional light's intensity and angle, providing a clear visual cue of day/night cycle progression. *(This "mini solar system" is visible only in the editor, meant to help developers visualize and orient the sky system. It can be toggled via Debug Parameters.)*
- **Daytime SkySphere Option:** Added an option to enable the SkySphere (sky dome) during daytime. Previously the starry sky sphere was only visible at night; now developers can choose to show it in daylight if desired (for stylistic or debugging purposes).

⏱️ Time & Date System Improvements

- **HourDuration Renamed:** The `HourDuration` variable has been renamed to **`HourLengthInSeconds`** for clarity. This represents the length of an in-game hour in real seconds.
- **Reverse Time Flow:** If `HourLengthInSeconds` is set to a negative value, time will now run **backwards**. This allows reversing the time progression (e.g. making the sun move in reverse).
- **Clamped Time Values:** The `StartHour` value is now clamped between **0** (min) and **DayDuration** (max), preventing invalid start times. Likewise, `DayDuration` cannot be set below **0** anymore.
- **Higher Time Precision:** Improved the precision of the day-night cycle. You can now use extremely small values for `HourLengthInSeconds` (up to 7 decimal places, e.g. 0.0000001) and the system will progress smoothly without issues.

- **Sun Movement Mode Renamed:** In the SunMovement settings, the mode previously labeled “Stationary” has been renamed to **“Static”** for consistency.
- **Initial Date Validations:** Enforced minimum values of **1** for `MonthQuantity`, `StartDay`, and `StartMonth`. This prevents starting with “zero” months or days, ensuring the date is always valid (e.g., there is no 0th day or 0th month).
- **Daily Sun Travel Output:** The `GetActualTime` function now includes a new output parameter **`DailySunTravel`**, which provides the sun’s travel angle for the current day (this was formerly available as `SunDegrees` in AdditionalVariables).

🌡️ Temperature System Enhancements

- **Multiple Temperature Units:** The weather system now supports temperature in **Celsius, Fahrenheit, or Kelvin**. In the Weather Config, a new dropdown allows you to select your preferred unit. All temperature-related settings and displays will use the chosen unit.
- **Constant Temperature Toggle:** Added a button (toggle) to **lock the temperature** at a constant value. You can set this fixed temperature in any of the supported units. When enabled, the temperature will remain fixed regardless of weather changes or time.
- **Temperature Interfaces:** New interface functions allow you to retrieve the current temperature in any of the three units (°C, °F, K) via Blueprints or code. This makes it easy to display or use the temperature value as needed.
- **Day/Night Indicator:** The system now provides an interface call to **check if it’s day or night**. This boolean can be accessed to, for example, trigger events when night falls or day breaks.

📘 UI and Blueprint Updates

- **Time Widget Renamed & Simplified:** The time display widget **`BP_HoraV2`** has been renamed to **`WBP_TimeDisplay`**. Its internal logic has been simplified by utilizing new SkyScape interfaces (for time and day/night status) to remove redundant calculations. The widget now more efficiently displays the current time without generating blueprint warnings.
- **Interfaces vs. AdditionalVariables:** The internal struct `SkyScape AdditionalVariables` is now hidden from external access. All values that were previously fetched from this internal struct (e.g. current sun angle, current temperature, etc.) are now accessible through clean **interface functions**. This encapsulation improves modularity and prevents direct misuse of internal data.
- **Blueprint Function Improvements:** The blueprint functions **GetCurrentParameters** and **GetReference** have been improved to eliminate log warnings. Calling these no longer produces warning messages, resulting in a cleaner experience when working with SkyScape in Blueprints.
- **Random/Sequential Function Renames:** Renamed the weather cycling functions for clarity – what used to be `RandomWeatherChanges` is now **`RandomWeatherStyles`**, and `SequentialWeatherChanges` is now **`SequentialWeatherStyles`**. These better reflect that the system cycles through weather **styles/presets**.

🌦️ Weather Transition System Overhaul

- **Interruptible Weather Transitions:** The weather interpolation system has been completely overhauled. You can now change the target weather **at any time**, even if a transition is already in progress. For example, if a transition from Sunny to Cloudy was set to take 10 seconds and is halfway through, you can now command it to switch back to Sunny (or to any other style). The system will smoothly interrupt the Cloudy transition and interpolate from the current state toward the new target weather. This makes weather changes much more dynamic and responsive.
- **New “Weather Transition Mode”:** Introduced a configurable mode to control how new weather targets are selected during automatic cycling:
  - **Disabled:** No special selection – the system will always interpolate to the **current** EditCurrentStyle (as it did in previous versions). Use this if you want to manually set the next weather style each time.

    - **Random:** On each transition, pick a **random weather preset** to interpolate to. (If no presets are provided, it defaults to the current EditCurrentStyle, same as Disabled mode.)
    - **Sequential:** Cycle through a predefined **sequence of weather presets** in order. (If no sequence is provided, it defaults to using the current EditCurrentStyle, as before.)
- **New "Weather Interval Mode":** Added a setting that determines what the system does *after* a weather transition completes, i.e. how it waits before starting the next change:
    - **Paused:** After a weather change finishes, the system will do nothing until explicitly restarted. It requires a manual call to **StartAgainSkyScape** each time a cycle ends. This is useful if you want to step through weather changes manually.
    - **Fixed Delay:** The system will wait a duration equal to **TimeToChangeWeather × Fixed Delay** before beginning the next weather transition; for example, if **TimeToChangeWeather** is 10 s and **Fixed Delay** is set to 2, the system will wait **10 s × 2 = 20 s** before starting the next interpolation.
    - **Random Interval:** The system will wait a **random amount of in-game time** before the next weather change. You provide a minimum and maximum interval (in hours), and after each transition the engine will pick a random value in that range for the delay. For example, if the range is 2 to 7 hours and a transition ends at 13:45 in-game time, it might pick 5 hours – meaning the next weather change will start at around 18:45 that day.
- **Random & Sequential Preset Lists:** The Random and Sequential weather change functions have been extended to accept **preset lists** of weather styles. This means you can prepare a list of weather presets to be used for random selection or sequential cycling, rather than relying only on the active EditCurrentStyle. If these lists are empty, the system simply uses EditCurrentStyle as a fallback (the original behavior).
- **Debug Fixed Delay Removed:** The old debug parameter for "fixed delay" has been removed, since the new **Weather Interval Mode** covers that functionality in a more robust way. Toggling the old debug option now only prints a debug message (it has no effect on the actual simulation).

🛠️ Optimizations & Bug Fixes

- **Optimized Code:** Numerous code optimizations were implemented under the hood to improve performance and efficiency of the SkyScape system.
- **Crash Prevention:** Added several validation checks to prevent potential crashes caused by out-of-range values or incorrect usage. The system now safeguards against invalid inputs (like the date and time clamps mentioned above) to ensure stability.
- **SkyLight/SkyAtmosphere Visibility Fix:** Fixed an issue where the SkyLight and SkyAtmosphere components were not being hidden when they were toggled off. They will now correctly become invisible/disabled when deactivated via the SkyScape settings.
- **Directional Light Initialization:** Adjusted the initialization of the Sun and Moon **Directional Lights** when they start deactivated. Now, if you start with the sun turned off (nighttime start), the sun's light intensity will begin at 0; similarly, if starting with the moon disabled (daytime start with no moonlight), the moon's light intensity begins at 0. This prevents unwanted light bleed from a supposedly disabled sun or moon at start.
- **Online Component Packaging Fix:** Resolved an issue with the SkyScape online component that was causing packaging errors when the component was present in a level. You can now include the network replication object (online component) in your scene and package the project without issues.
- **Removed Obsolete Function:** The internal function for auto-finding a Landscape in the level has been removed, as it became obsolete and was no longer needed.
- **General Stability:** Many minor tweaks and fixes are included to make the system more robust overall, based on user feedback from previous versions.

SkyScape 1.9.1 – Patch Notes(25/07/2025)

Added:

- **Probability-based Transition Mode:** Introduced a new *Weighted* transition type that allows you to assign probabilities to a set of weather presets. During climate transitions, SkyScape will automatically select one of these presets based on the defined probabilities.
- **Getter and Setter Functions:** Added new accessor methods to retrieve and modify SkyScape variables safely, improving the ability to configure the system via code or blueprints.

Changed:

- **Internal Variables Encapsulation:** Certain internal SkyScape variables are now hidden (made private) to prevent improper use or unintended modifications. This change enhances stability by restricting direct access to these variables.
- **Renamed Functions for Clarity:** Several SkyScape functions have been renamed to better reflect their behavior:
  - `applystyleSkyScape` → `ExecuteSkyReconfiguration` (immediately apply the current weather style)
  - `StopSkyScape` → `PauseSimulation` (pause the SkyScape simulation)
  - `StartAgainSkyScape` → `ResumeSimulation` (resume the paused simulation)
  - `SaveSkyScape` → **SkyScape Save System** (saving is now handled by the new **SkyScape SaveSystem** – use the `SaveSkyScapeToSlot` function)
  - `savedeleteSkyScape` → `DeleteSkyScapeSaveSlot` (delete a saved state from a slot)
  - The function to check for an existing save slot is now called `DoesSkyScapeSaveSlotExist`

Removed:

- **SkyScape Interface:** The SkyScape interface has been removed from this version, as it is no longer needed. All interactions should now be done through the updated SkyScape API and systems described above.

SkyScape 1.9.2 – Patch Notes (26/07/2025)

Changed:

- **Interface Improvements:** Hidden the **Rendering** and **HLOD** parameters of **SkyScape**, along with other settings, to ensure a clean plugin interface in **Unreal Engine 5.6**.

SkyScape v1.9.3

This release brings a series of API renames for greater clarity, a powerful new time-control feature, smoother online synchronization, a critical transition bug fix, and various code cleanups to eliminate compile-time warnings.

🚀 API Renames & Consistency

- **EditCurrentStyle → CurrentWeatherStyle**
  - All references and Blueprint pins have been updated to use the new name—reflects that this struct now exclusively defines weather parameters.
- **FStartConfig → FSystemConfiguration**
  - Renamed to better convey its role as the global configuration container for SkyScape's core systems (time, season cycles, default styles).

🪲 Bug Fixes

- **Transition Interruption**
  - Fixed an edge case where interrupting one weather transition with another (for example, a transition that disables clouds) would leave clouds permanently hidden. Now any overlapping transition resets and reapplies cloud visibility correctly.

✨ New Features & Improvements

- **SetTime**
  - Introduces a new `SetTime(float NewTime)` function, allowing developers to jump to any hour/minute at runtime.
  - Exposed in both standalone and networked (online) modes.
- **SkyScapeSynchronize Optimization**
  - Under-the-hood improvements to `SkyScapeSynchronize` drastically reduce latency and jitter when syncing world time across multiple clients—even over high-latency connections.

🛠️ Code Cleanup & Warning Resolution

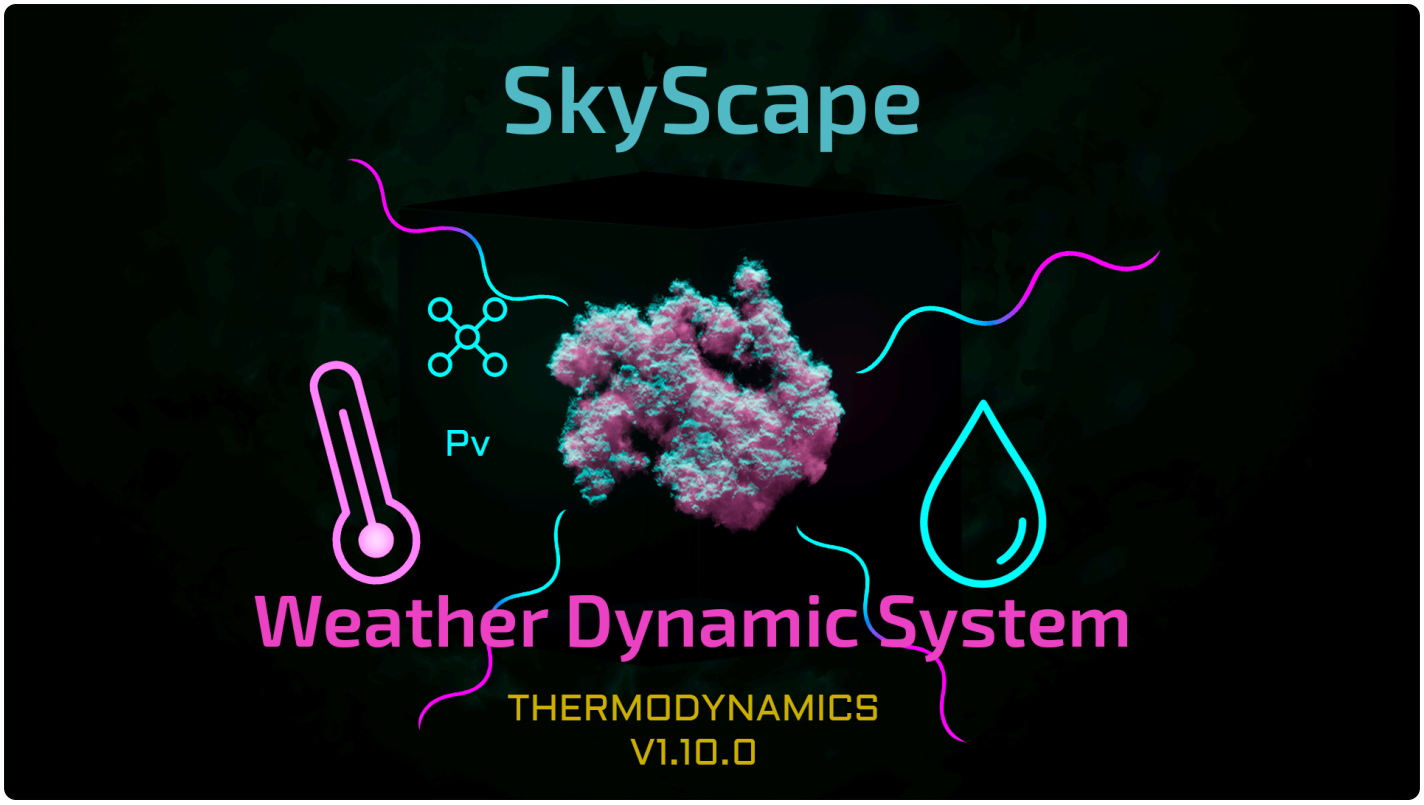
- Refactored several internal data structures and Blueprint-exposed settings to resolve lingering compile-time warnings in both engine and plugin modules.
- Improved struct alignment and removed deprecated properties to ensure a clean, warning-free build with Unreal Engine 5.6.

Thank you for using SkyScape! We recommend updating to v1.9.3 to take advantage of these enhancements and ensure maximum stability.

SkyScape V1.9.4 Patch Notes

- **Setters hidden**: Now only callable via the Blueprint Function Library.
- **Presets hidden** on simulation start.
- **Private category names renamed** for clarity.
- **New utility functions added**: retrieve day, month, year; load a save slot; check if a save slot exists.
- **Validation added**: “Day in One Season” now has a minimum of **1**.
- **Save system bug fixed** that was preventing saves from working.
- **Online functions added**.
- **SkyScape Functions reorganized** into categories for better discoverability.

SkyScape V1.10.0 Release Notes (14/08/2025): Thermodynamics



SkyScape 1.10.0 refines the core weather simulation and streamlines day-to-day authoring. This update introduces a dual-mode temperature model, richer sun/location controls, safer date validation, preset quality-of-life improvements, and a revamped API for getters/setters.

🌡 Thermodynamics Overhaul

- **Temperature calculations updated:** added **Humidity**, **Vapor Pressure (Pv)**, and **Saturation Vapor Pressure (Pvs)** systems.
- **Temperature Mode** (Weather Config): new dropdown to choose **Simple** or **Complex**.
  - **Simple:** preserves legacy behavior.
  - **Complex:** exposes advanced temperature options for detailed tuning.

☀ Sun & Location Controls

- Added **Longitude**, **Time-Zone Offset**, and **Daylight-Saving Time** options to the Sun settings.

📆 Safer Initial Date

- **Start Month** is clamped to the number of defined months.
- **Start Day** is clamped to the valid range for the selected month.

🎚 Presets & Authoring

- **Auto-refreshing presets:** if a preset is placed/assigned, changes to it now update the actor automatically.
- **Preset Notes field:** each preset includes an internal **Text** note to guide users (e.g., “recommended: Night + RealisticClouds”).
- Internal SkyScape-only functions are **hidden from Blueprints** to reduce noise and accidental misuse.

🧩 API & Blueprint UX

- SkyScape functions updated; **getters and setters fully revamped**.
- Many new **setter** calls to adjust individual parameters directly—**no more breaking structs** just to tweak one value.

- Function library/categories refined for clearer discoverability.

🖼 **UI**

- **Widgets updated** to reflect the new modes and options.

🗂 **Project Hygiene**

- **Dependency folders reorganized** for cleaner structure.

✅ **Notes for Upgrading**

- Because getters/setters changed, projects may require **relinking Blueprint pins** or updating calls to the new setters/getters after install.
- If you previously relied on internal functions from Blueprints, use the public API equivalents now exposed in the function library.

SkyScape V1.10.1 Patch Notes

- **Bug Fixes**

SkyScape V1.10.2 Patch Notes

- **Bug Fixes**

SkyScape V1.10.3 Patch Notes

- **Code Improvements**

SkyScape V1.10.4 Patch Notes

- **Bug Fixes**

SkyScape V1.10.5 Patch Notes

- **Bug Fixes**
- **Event Dispachers:** Event bindings were added to the SkyScape API to retrieve parameters, such as the moment when it switches from night to day.

-Technical Details

- Fully compatible with Unreal Engine 5.0-5.6.
- Modular design for seamless integration into new and existing projects.
- Includes documentation, tutorial videos, and a free showcase for hands-on testing

SkyScape – Future Roadmap

We're thrilled to share what's coming next for SkyScape! Here's a sneak peek at the upcoming features and improvements:

1. **Performance Optimizations**
   Continued improvements to ensure SkyScape remains efficient, even in large and complex projects.

Videos Tutorials(English)

Playlist en Youtube: Open link

Download SkyScape: Open link

SkyScape: Installing the Plugin and Placing SkyScape on the Map

SkyScape: Basic Functions

SkyScape: Weather Sequences and Random Weather

Videos Tutorials(Spanish)

Playlist en Youtube: Open link

Download SkyScape: Open link

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

SkyScape: Materiales Dinámicos según el Clima y Estaciones

SkyScape: Consejos Finales y Optimización de SkyScape

SkyScape: Packaging & Publicación

SkyScape Discord

Discord Link: https://discord.gg/CEWnCSJyEh

Problems SkyScape Solves

Download SkyScape:

SkyScape is a **comprehensive solution** that handles many essential **climate and time-management elements** of game development, letting you focus on **core gameplay**. Control **random or fixed weather cycles**, **sync with real time**, and easily configure **custom seasons**.

Product Features

SkyScape is a versatile product designed to enhance your experience by offering a unique approach to managing weather and climate settings.

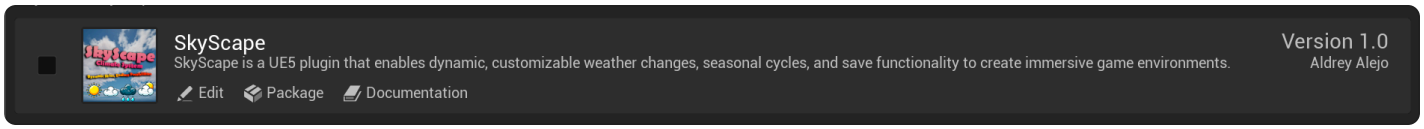
- **Date and Time Management**: Easily manage dates and times within the product.
- **Realistic Weather Changes**: Allows for realistic changes in weather, including adding or removing seasons.
- **Climate Saving System**: Includes a system for saving different climates.
- **Random Climate Changes**: Automatically changes climates randomly based on elapsed time.
- **Sequential Climate Changes**: Sequentially changes climates based on elapsed time, creating a dynamic environment.

Write that its implementation is very simple and that the product weighs less than 75MB.

How to integrate SkyScape with your project

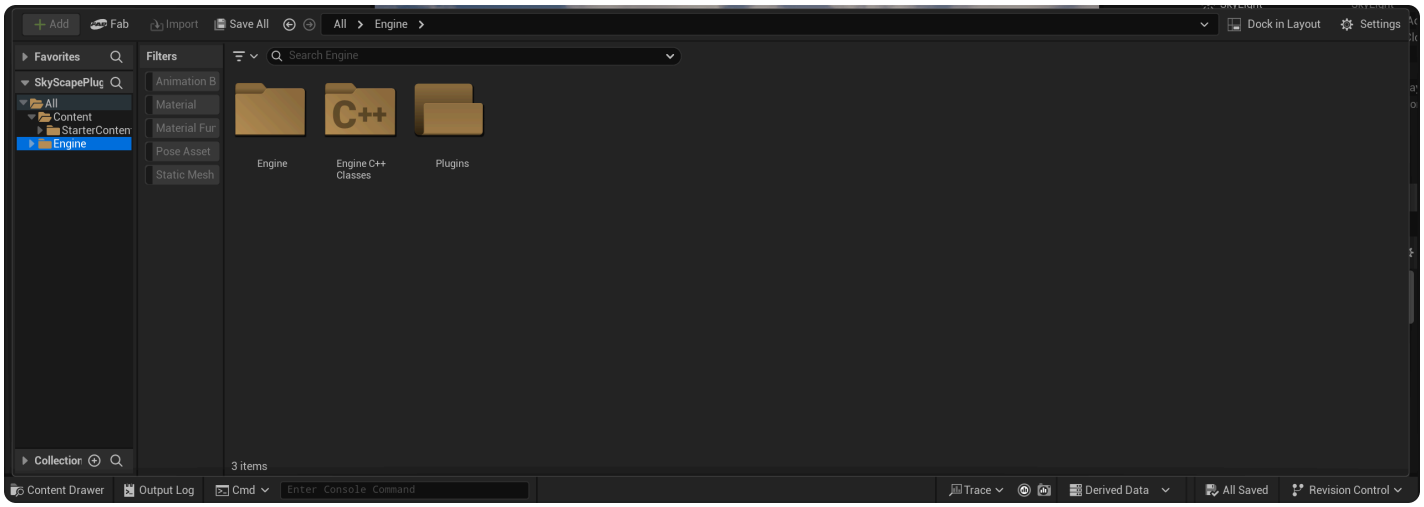
Download SkyScape:(fabLink)

To integrate **SkyScape** into your project, the first thing we'll do is enable the **plugin**:



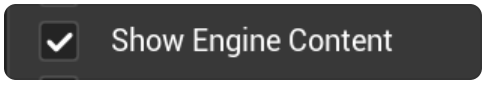
After enabling it, we **restart Unreal**.

Open The **Content Drawer**.

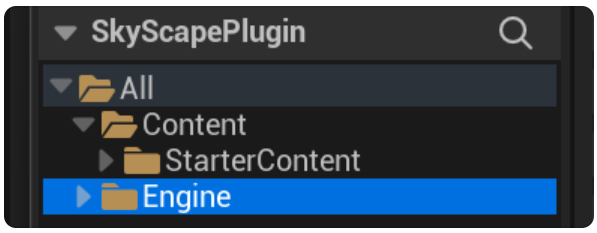


Go to **settings.**

Activate **Show Engine Content**.



Open the **Engine folder**.



Open the **plugins folder**.

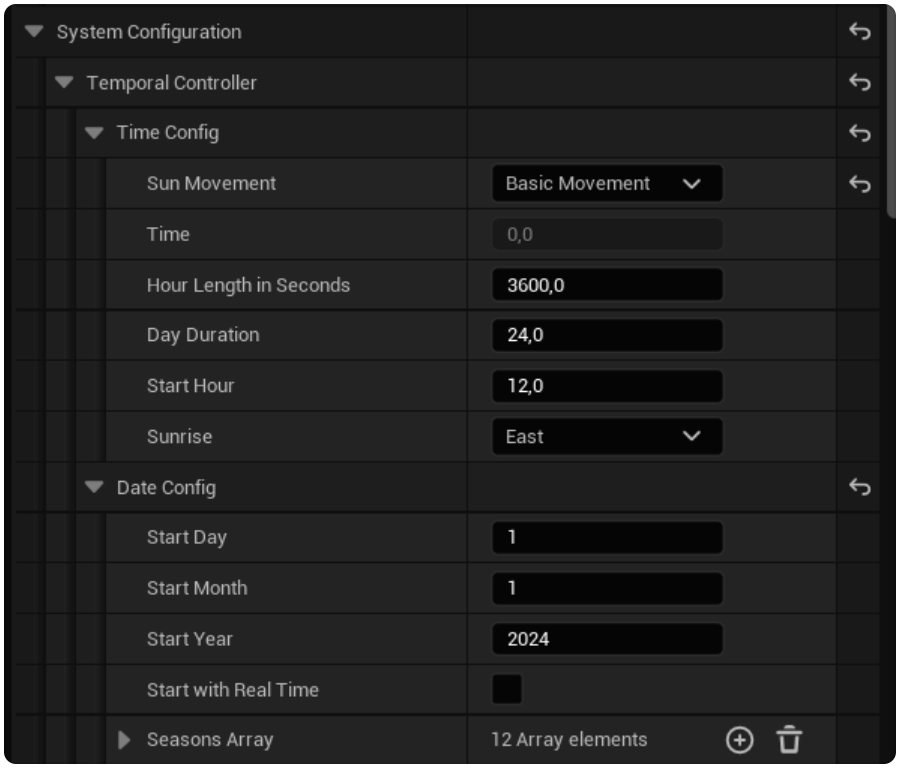
Search for **SkyScape C++ Classes**.

Drag and drop **SkyScapeV2** into your map.

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

Changing the Time and Synchronizing with Real Life

SkyScape can synchronize its in-game clock and date to **real-world time**. This means the simulated sun, moon, and sky can start at the actual current time and date of the player's system. This feature is great for projects that want the environment to reflect real life (for example, an application that always shows the current time of day).



Enabling Real-Time Start

To have SkyScape use the real world's time and date at runtime, follow these steps:

- In the SkyScape actor's Details panel, locate **Date Configuration** under the **Temporal Controller** settings.
- Enable **"Start With Real Time"** (`bStartWithRealTime`). This is the critical setting that tells SkyScape to use the system clock on game start.
- (Optional) Under **Time Configuration**, ensure **Sun Movement** is set to an option that allows time progression (either **Basic Movement** or **Advanced Movement** – more on these below). If you leave Sun Movement as **Static**, the sky will be set to the current real time once, but then remain fixed.

With *Start With Real Time* enabled, SkyScape will, during BeginPlay, fetch the current date and time from the system and apply it:

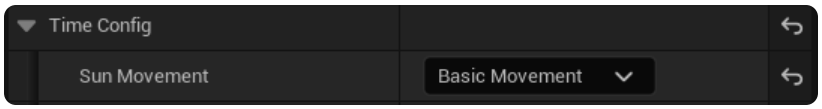
- **Current Date:** The plugin reads the real-world date (day, month, year) and overrides the Start Day/Month/Year to these values.
- **Current Time:** It also calculates the current time of day (including hour and fractional minutes) and sets that as the Start Hour. For example, if the player begins the level at 3:30 PM local time, SkyScape will set the in-game start hour ~15.5 (15 hours + 30 minutes).
- These values then propagate to SkyScape's simulation – meaning the sun's angle, moon phase, and sky appearance adjust to what they would be on that date and time.

Tip: You can still adjust the starting date/time manually if needed. The *Start With Real Time* setting simply overrides the Start Day/Time on BeginPlay. If it's off, SkyScape uses whatever Start Day/Month/Year and Start Hour you have set in the details.

Real-Time Progression vs. Static Snapshot

After the initial synchronization, you have two choices for how time proceeds:

- **Real-Time Progression:** If you want the in-game time to continue advancing in step with real time (1 second per real second), configure the time scaling accordingly. By default, **Hour Lenght In Seconds** in SkyScape is 3600 seconds, meaning one in-game hour lasts 3600 real seconds – exactly one real hour. Combined with **Day Duration = 24** hours, this yields a 24-hour day/night cycle that matches real time. As long as Sun Movement is Basic/Advanced (not static) and you haven't modified the time scale, the sky will now move in sync with the real world clock. In other words, if you play your app at 6 PM and leave it running, by 7 PM real time the sky will have progressed to sunset or dusk naturally.
- **Static Real Time Snapshot:** If instead you use **Static** sun movement, SkyScape will set the sky to the current time once and then freeze the sun's motion. The sky won't change unless you manually adjust it. This is useful if you want a scene to always start at the real current time but not advance (for instance, always show "right now's" sky at the moment of launch). You might use this mode for an AR/VR experience where the user's current sky is shown as a backdrop.

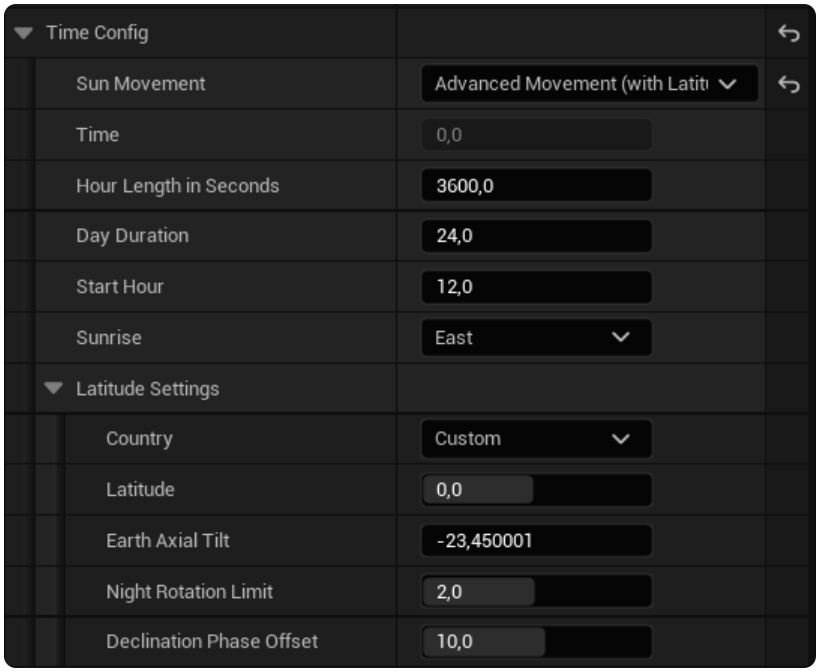


In summary, enabling *Start With Real Time* gives you an initial alignment. To maintain synchronization, ensure the day length and time scale are appropriate for real time, or choose static mode for a one-time set.

Location and Sun Position (Advanced Movement)

If you require the sun and moon positions to be astronomically correct for real-world time, you should use **Advanced Movement (with Latitude)** for the sun's movement mode:

- In **Time Configuration**, set **Sun Movement** to **Advanced Movement (with Latitude)**. This mode factors in the date, time, and your geographic location (latitude) to calculate sun angle and day/night length.
- Select your **Country** or set a custom latitude/longitude. SkyScape provides an **ECountry** list with many countries; choosing one will auto-set the latitude (and possibly time zone offset) for that region. If you set **Country = Custom**, you can input your exact latitude in the **LatitudeSettings** (in degrees).
- With Advanced Movement, the simulation will account for seasonal changes in sunrise/sunset. For example, if it's winter and you're at a northern latitude, the daylight hours will be shorter. The *Start With Real Time* feature will ensure it's also the correct date, so you get the appropriate season.



Note: Time zone considerations – SkyScape uses your system clock directly. Ensure the system time is correct for the local time zone. If you want the sky to reflect a different time zone, you would need to offset the Start Hour manually (SkyScape doesn't have a built-in time zone offset parameter; it assumes local time).

Best Practices for Real-Time Sync

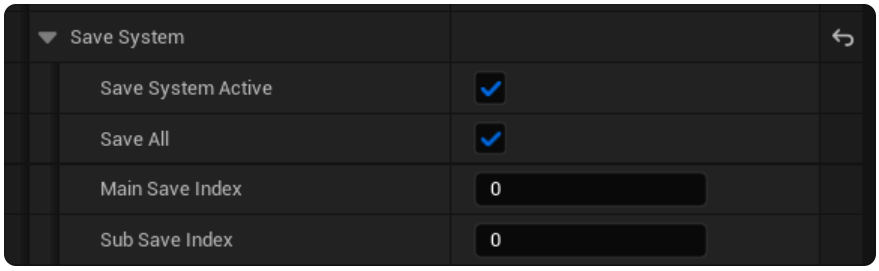
- **One SkyScape Actor:** Typically, you'll have one SkyScape actor in the level. Setting it to real time affects the whole level's lighting and sky. You wouldn't mix multiple sky actors with different times.
- **Testing:** When testing in editor, remember that *Start With Real Time* will use your PC's clock. So hitting Play at different times yields different sky conditions. This can be a quick way to preview sunrise, noon, night, etc., by just changing your system clock or waiting for a specific time.
- **Game Design Considerations:** Real-time sync is immersive (the game world matches real world), but it also means the player cannot speed up or slow down time. In some cases, designers prefer an accelerated cycle so players experience day and night within a short play session. Decide what's best for your project. You can always disable real-time after the start or adjust **HourLenghtInSeconds** if you want to drift away from real time after initialization.
- **Advanced Movement Benefits:** Even if you don't need true real-world sync, using Advanced Sun Movement with a correct latitude can make your sky behave more realistically over in-game days (with correct sun angles and moon phases for the date). The *Start With Real Time* just gives it a correct starting reference point.

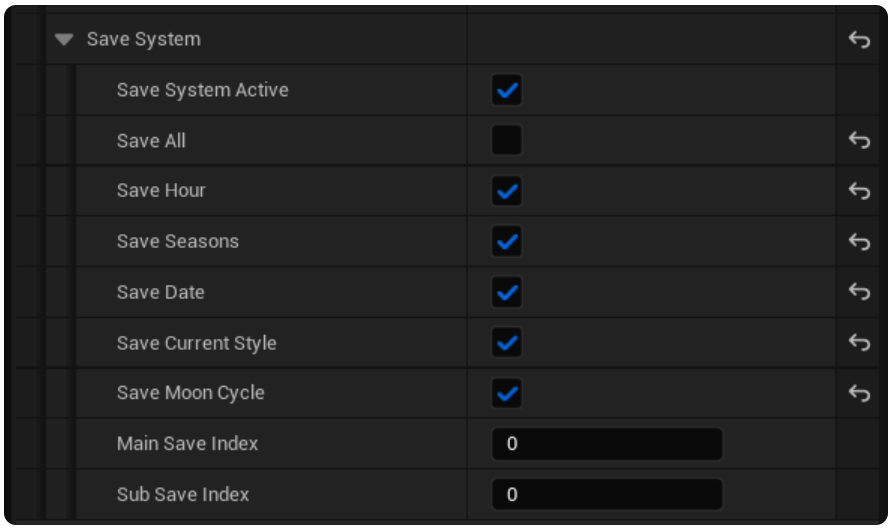
By synchronizing with real life, SkyScape can provide an extra layer of realism – imagine an architectural viz app where the sky outside the window matches the actual sky when you open it, or a game that ties into the player's local time for events. It's a powerful feature when used thoughtfully.

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

Save System

SkyScape includes an **advanced save system** that lets you preserve the weather state, time of day, date, and other settings between play sessions. This system uses a slot-based approach with a **Main Slot** and a **Sub-Slot** to organize multiple saved states. Key features and usage steps are outlined below.





Enabling and Configuring the Save System

To use the save system, first enable it in the SkyScape actor's settings:

- **Enable Save System:** In the SkyScape actor Details panel under **Save System**, set "**Save System Active**" (`bIsSaveSystemActive`) to **true**. This tells SkyScape to load a saved state at game start (if one exists).
- **Choose Slot Indices:** Specify a **Main Save Index** and **Sub Save Index** (both integers ≥ 0). These two numbers identify the save file and the entry within that file. For example, Main Index 0 with Sub Index 0 could be your primary save. (You can use different Sub-Slots to store multiple weather profiles or checkpoints under one Main slot file.)
- **Select What to Save:** Check the options for which aspects of SkyScape to persist:
  - **Save Hour** (`bShouldSaveHour`): Preserve the current time of day (hour/minute) on save.
  - **Save Date** (`bShouldSaveDate`): Preserve the current date (day, month, year).
  - **Save Seasons** (`bShouldSaveSeasons`): Preserve seasonal progress (ensures the same season continues after load).
  - **Save Current Style** (`bShouldSaveCurrentStyle`): Preserve the active weather style and all its parameters (cloud density, rain intensity, etc.).
  - **Save Moon Cycle** (`bShouldSaveMoonCycle`): Preserve the current moon phase/day in the lunar cycle.
    Tip: Enable all relevant options so that your world looks exactly the same when loaded. If an option is unchecked, that aspect will reset to the default from the SkyScape actor's setup on load.

Automatic Loading on BeginPlay

When the Save System is active, SkyScape will automatically attempt to **load the saved state** at the start of the game (BeginPlay). It uses the Main/Sub slot you configured:

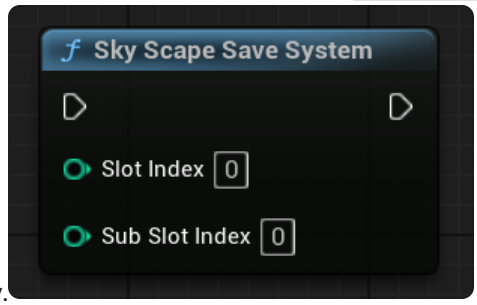
- **On Game Start:** If a save exists in the specified slot, SkyScape will load it immediately during BeginPlay. This restores the sky to the last saved time, date, weather, etc. You don't need to manually call a load function – it's handled for you.
- **If No Save is Found:** If there is no save data for that slot, the plugin will simply use the SkyScape actor's default configuration (no errors – it just means this is the first run or the slot is empty).

Note: Make sure to use the same slot indices for saving and loading. For example, if you intend to always continue where the player left off, keep the Main/Sub indices constant (e.g. 0 and 0) across sessions.

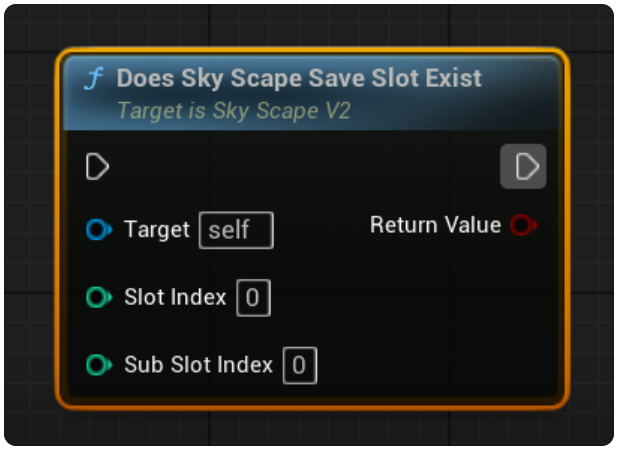
Saving the Sky State via Blueprint

You can save the current SkyScape state at any time during gameplay using Blueprint nodes:

1. **Call the Save Function:** Use the `SaveSkyScapeToSlot` Blueprint node (from the SkyScape actor). Provide the same Slot Index and Sub Slot Index you chose earlier. This will capture the current sky configuration and write it to the SaveGame slot. Internally, SkyScape will update or create a file named `"SkyScapeSaveSlot_<SlotIndex>"` and store the data in the specified SubSlot entry.



2. **Optional – Confirmation:** You can use `DoesSkyScapeSaveSlotExist` (returns a boolean) to check if a save slot already exists before or after saving. This is useful if you want to, say, grey out a "Continue" button unless a save is present.



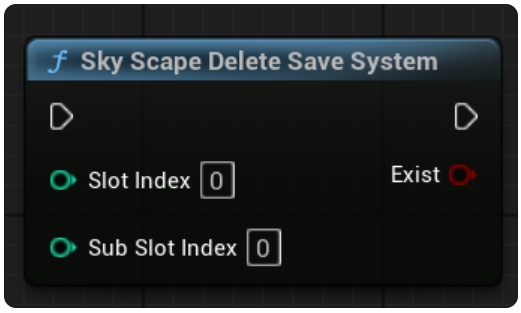
3. **Multiple Saves:** The save system supports multiple entries. You might use Main Slot 0 for all game saves and vary SubSlot (0, 1, 2…) for different save points or profiles. Just remember to use the matching pair when loading.

There is no separate "**Load**" Blueprint node because loading is handled automatically on **BeginPlay** when the system is active. If you need to implement a manual in-game load (for example, a menu that loads a different save), the typical approach is to change the SkyScape actor's **Main/Sub indices** to the desired slot and then travel or reload the level, triggering BeginPlay again. (Ensure **bIsSaveSystemActive** remains true.)

Deleting Save Data

SkyScape also provides a way to delete saved weather data via the `DeleteSkyScapeSaveSlot` Blueprint function:

- When called with a **Main Slot and Sub Slot**, the system will remove that entry. If it was the only **sub-slot** in the file, the entire save file will be deleted from disk.
- The function returns true if the deletion succeeded. You might use this to implement a "**Clear Weather Save**" option or to manage old saves.



Best Practices for the Save System

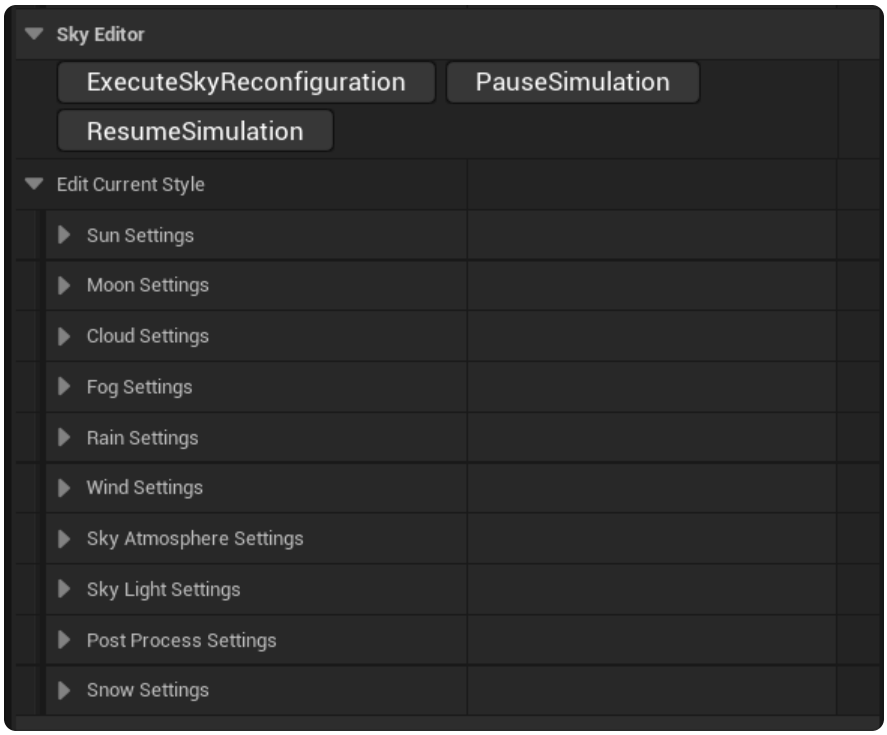
- **Consistent Indices:** Decide on a scheme for your save slots. For a single persistent world, using (0,0) for everything is simplest. For multiple profiles or levels, you can assign different sub-slots or main slots as needed.
- **Save Timing:** Call `SaveSkyScapeToSlot` at logical points (e.g., when the player quits, or at checkpoints). The save operation is quick since it only stores SkyScape data.
- **Loading Customization:** The automatic BeginPlay load will only happen once (when the SkyScape actor is initialized). If you disable the Save System (set it inactive), SkyScape will always start fresh. This can be useful if you want some levels to always use a fixed weather setup while others continue from a save.
- **Data Persistence:** The saved SkyScape state is stored using Unreal's SaveGame system, meaning it persists between runs of the game. Ensure you handle versioning or resetting of saves if you drastically change your SkyScape setup in an update.

By using the Save System, you can create experiences where the game's environment truly persists – players can return to a game and see the same time of day, weather conditions, or even moon phase as when they left off, providing continuity and immersion.

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

Default Weather

To set a **default weather** that does not change over time, first, we need to **enable** the option '**GlobalPause**' and then scroll down to the **SkyEditor** section, where we will go into '**Edit Current Style**':



The **Edit Current Style** section contains the parameters for configuring the current weather settings. Each parameter allows for detailed customization of various environmental elements. Here's a breakdown of how it works:

1. **Activation Button**
   - Each section (e.g., Sun Settings, Cloud Settings) includes a toggle button (e.g., **Sun Active**, **Cloud Active**) to enable or disable that specific element in the weather system.
   - If unchecked, the corresponding element will not be visible or active in the environment.
2. **Presets**
   - Each section offers a **preset option** with predefined settings for the selected element. These presets include:
     - **Low**: Minimal effect for the selected parameter.
     - **Medium**: Balanced settings for moderate impact.
     - **High**: Maximum effect for the element.
     - **Custom**: Allows full manual customization of the parameter values.
3. **Custom Settings**
   - When **Custom** is selected in the preset dropdown, the **Custom Settings** section becomes active. This allows you to manually adjust individual parameters (e.g., intensity, color, size) to fine-tune the weather system according to your needs.
   - These settings are particularly useful for creating a **default weather** that aligns with your game's style or requirements.

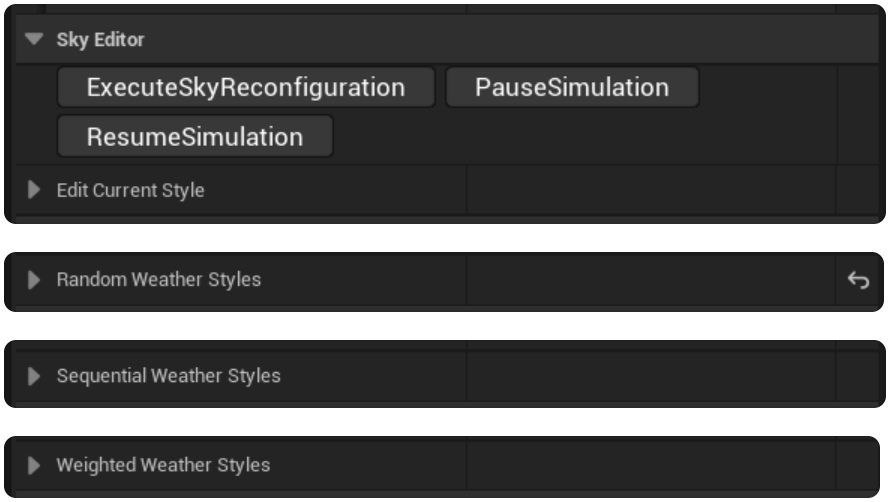
Example Walkthrough

- If you want to create a default sunny weather:
  a. Go to **Sun Settings**.
  b. Enable **Sun Active**.
  c. Set the **Sun Preset** to **Custom**.
  d. Expand **Custom Settings** and adjust parameters like sun intensity, direction, and color.
  e. Repeat similar steps for **Cloud Settings**, **Rain Settings**, etc., adjusting each element as needed.

SkyScape: Instalando el Plugin y Colocando SkyScape en el Mapa(22/4/25)

Sequential Weather and Random Weather

To create weather changes, first open the **Weather Transition Mode** dropdown (default: **Disabled**, which simply copies and interpolates your **Edit Current Style** values). When you select **Sequential**, **Random** or **Probability**, the matching **Sequential Climate Change**, **Random Climate Change** or **Probability Climate Change** panel will appear beneath **Edit Current Style**.



These options act as a selector: for **Random** and **Sequential**, they let you choose between an array of **Edit Current Style** entries or an array of presets (whichever is more convenient). For **Probability**, it's a map pairing each preset with an integer probability— in recent UE5 versions this runs from 0 to 100; in older versions there's no hard limit, though it's best to stay within that range.

To change the modes, simply go to **System Configuration** and find the **Weather Transition Mode** option.

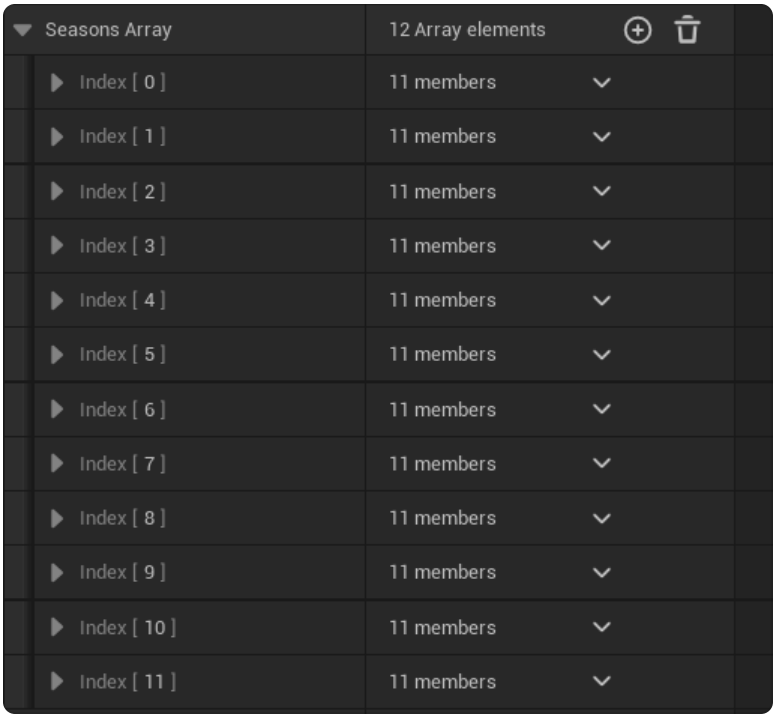
- **Random Changes**
  Random changes work by selecting one of the defined styles **at random** and applying it as the active weather. After a set amount of time, it will randomly pick another style and switch to that weather condition. This process continues indefinitely, creating a dynamic and unpredictable weather system.
- **Sequential Changes**
  Sequential changes follow the **order** in which the styles are defined. If you have three styles created, the system will start at **index 0**, then move to **index 1**, then to **index 2**, and finally loop back to **index 0**. The sequence will repeat in the same order, providing a more predictable and structured weather progression.
- **Weighted Changes**

  Weighted changes select weather styles at random according to assigned **probabilities (weights)**. Each style in your preset list has a **weight value** that determines how likely it is to be chosen: **higher-weight styles** appear more often, **lower-weight ones** less so. After each interval, the system performs a **weighted random draw** and applies the new style, then **repeats indefinitely**—resulting in a **dynamic but controllable weather mix**.

SkyScape: Climas Secuenciales y Climas Aleatorios

Seasons of the Year

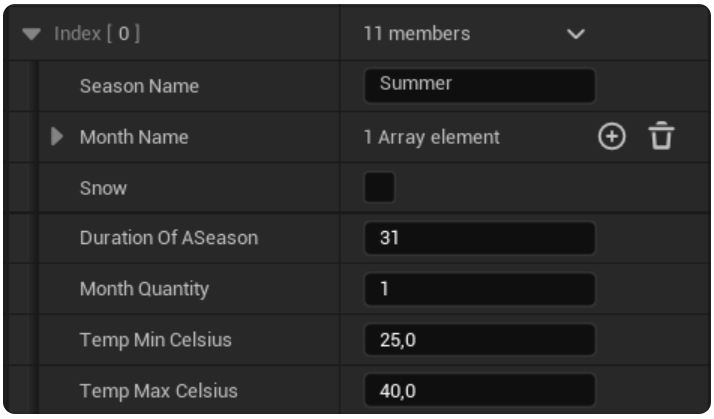
The seasons are defined in the **Sun Configuration** under the **Seasons** section. The first step here is to enable **Use Seasons** and then define the seasons within this array.



1. **Season Name**
   - **Description**: This defines the name of the season, such as "Spring," "Summer," etc.
   - **Usage**: Use it to label each season for easier identification in the array.
2. **Month Name**
   - **Description**: This array lists the names of the months included in the season.
   - **Usage**: Add the names of the months that belong to this season. For example, "January," "February," and "March" for Spring.
3. **Snow**
   - **Description**: This is a boolean that determines whether snow will be active during the season.
   - **Usage**: Enable this option if you want the season to include snowy weather effects.
4. **Duration Of A Season**
   - **Description**: Specifies the total number of in-game days the season will last.
   - **Important Note**: The value must be **divisible by Month Quantity** to ensure a whole number of days per month. For example, if the duration is 90 and the month quantity is 3, each month will last 30 days. Avoid values that result in decimals (e.g., 91 divided by 3 = 30.33 days).
5. **Month Quantity**
   - **Description**: Defines the number of months in the season.
   - **Usage**: Ensure this matches the number of months listed in the **Month Name** array.
6. **Temp Min (Minimum Temperature)**
   - **Description**: Sets the minimum temperature for the season.

- **Usage**: Define the lowest temperature expected during this season, such as 5°C for Spring.

7. **Temp Max (Maximum Temperature)**
   - **Description**: Sets the maximum temperature for the season.
   - **Usage**: Define the highest temperature expected during this season, such as 25°C for Spring.

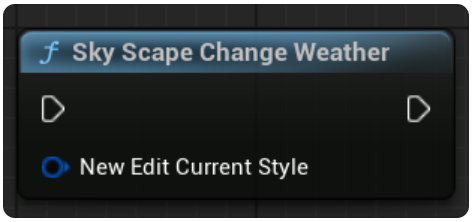


SkyScape: Gestionando las estaciones

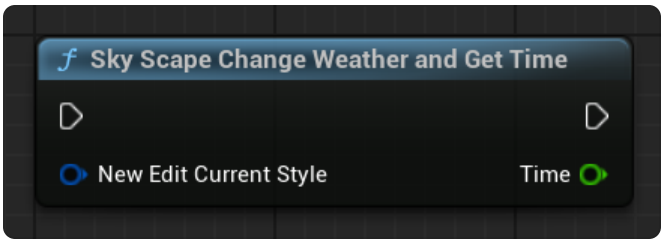
Change Weather Using Blueprints

In this section, you will see the **blueprints** included in **SkyScape**:

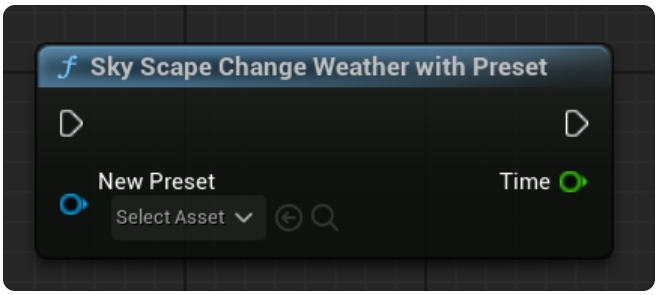
1. **SkyScape Change Weather**
   - **Description**: This function allows you to change the current weather in the SkyScape system.
   - **Usage**: Call this function to immediately set a new weather condition.



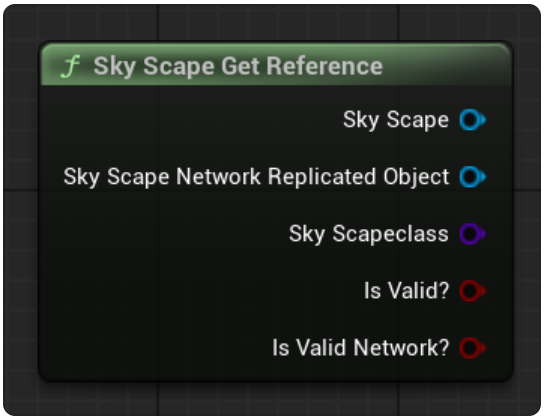
2. **SkyScape Change Weather And Get Time**
   - **Description**: This function changes the current weather and returns the time it will take to complete the transition.
   - **Usage**: Useful for calculating when the new weather will fully take effect.



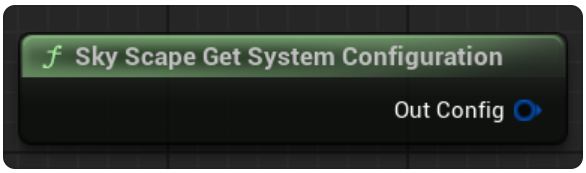
3. **SkyScape Change Weather And Get Time**
   - **Description:** Applies a SkyScape preset to transition from the current weather to the specified preset. Returns the time required to complete the transition.
   - **Usage:** Provide a SkyScape preset as input to change the current weather conditions to match that preset. The node outputs how long the transition will take.



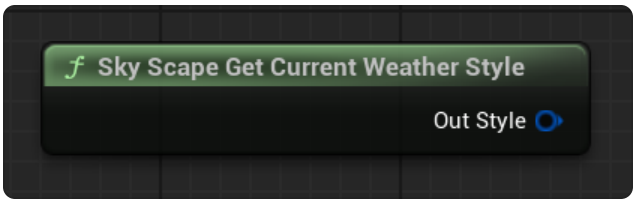
4. **SkyScape Get Reference**
   - **Description**: Provides a reference to the SkyScape actor in the current level.
   - **Usage**: Use this function to interact with SkyScape components and parameters.



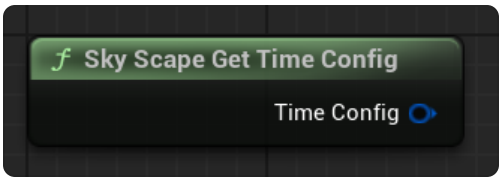
5. **SkyScape Get System Configuration**
   - **Description**: Returns the current SkyScape system configuration as a full struct.
   - **Usage**: Call it to read the state; break the struct to access fields. Use setter nodes to apply changes.



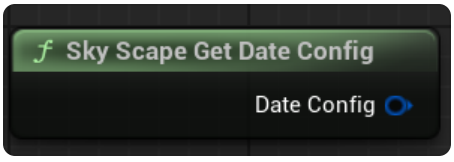
6. **Sky Scape Get Current Weather Style**
   - **Description:** Returns the currently active weather style/preset as a struct.
   - **Usage:** Call to read the active style; break the struct to inspect fields. Use setter nodes to change the weather.



7. **Sky Scape Get Time Config**
   - **Description:** Returns the Time Config sub-structure directly (the same one contained in System Configuration).
   - **Usage:** Call to read time settings without fetching the whole system config; break the struct to access fields.

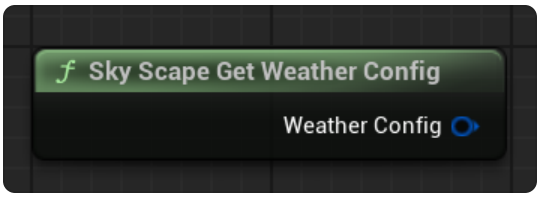


8. **Sky Scape Get Date Config**
   - **Description:** Returns the Date Config sub-structure directly (the same one contained in System Configuration).
   - **Usage:** Call to read calendar/date settings without fetching the whole system config; break the struct to access fields.

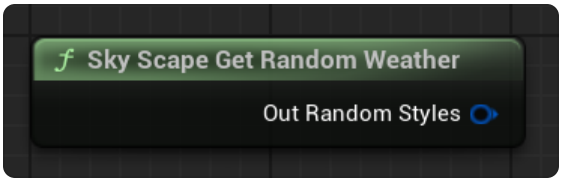


9. **Sky Scape Get Weather Config**

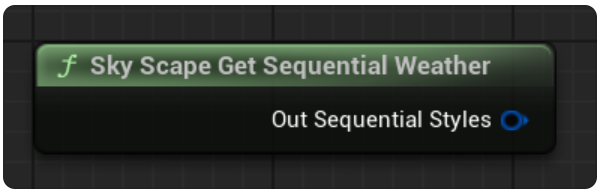
- **Description:** Returns the Weather Config sub-structure directly (the same one contained in System Configuration).
- **Usage:** Call to read weather settings without fetching the whole system config; break the struct to access fields.



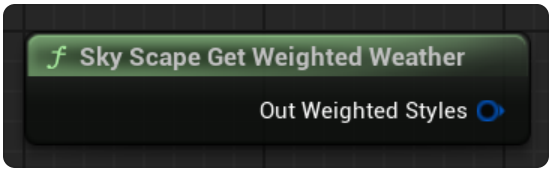
10. **Sky Scape Get Random Weather**
    - **Description:** Returns the array of random weather styles configured in SkyScape.
    - **Usage:** Call to read the pool used by Random transitions; iterate or pass it to your logic. To modify the pool, use **Sky Scape Set New Random Climates**.



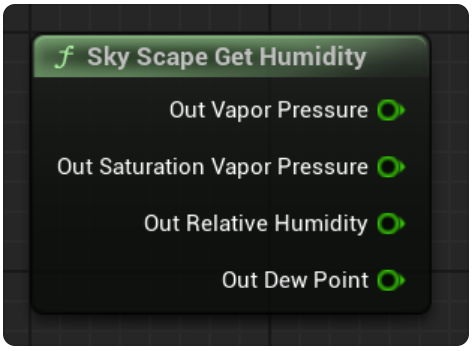
11. **Sky Scape Get Sequential Weather**
    - **Description:** Returns the ordered array used by the Sequential weather cycle.
    - **Usage:** Call to read the sequence; to modify it, use **Sky Scape Set New Weather Cycle**.



12. **Sky Scape Get Weighted Weather**
    - **Description:** Returns the weighted weather pool used by the **Weighted** transition mode (each entry includes its weight/probability).
    - **Usage:** Call to read or display the current weighted list; adjust weights in your presets or via the relevant setter nodes.

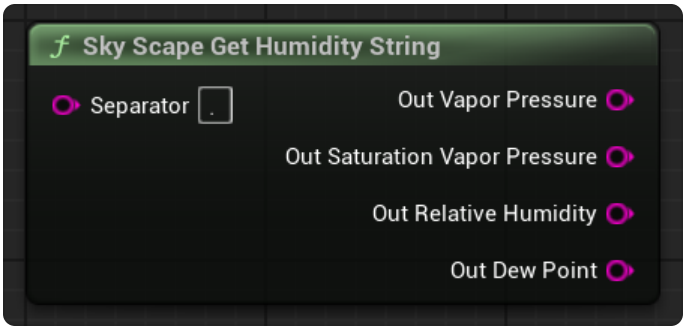


13. **Sky Scape Get Humidity**
    - **Description:** Returns the current humidity parameters as floats.
    - **Usage:** Call to read live humidity state for UI, logic, or logging.



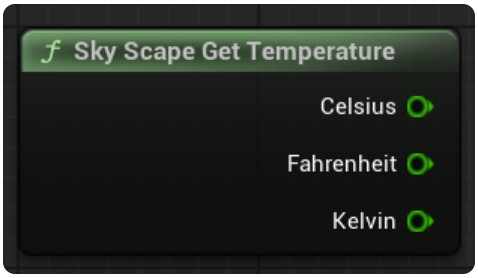
- **Outputs:**
  - **Out Vapor Pressure** — Pv (float, hPa).
  - **Out Saturation Vapor Pressure** — Pvs (float, hPa).
  - **Out Relative Humidity** — RH (float).
  - **Out Dew Point** — Dew point (float, uses current temperature unit).

14. **Sky Scape Get Humidity String**
    - **Description:** Returns current humidity parameters as **formatted strings**, using the chosen decimal separator and including units.
    - **Usage:** Set **Separator** to `"."` or `","`; call the node to get UI-ready text.



    - **Inputs:** **Separator** — Decimal delimiter to use (`"."` or `","`).
    - **Outputs:**
      - **Out Vapor Pressure** — Pv string (hPa).
      - **Out Saturation Vapor Pressure** — Pvs string (hPa).
      - **Out Relative Humidity** — RH string (%).
      - **Out Dew Point** — Dew point string (uses current temperature unit: °C/°F/K).

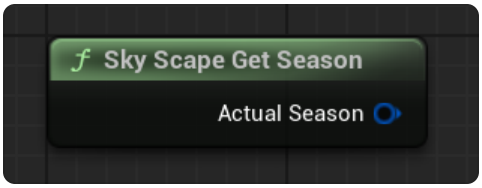
15. **Sky Scape Get Temperature**
    - **Description:** Returns the current air temperature as floats in three units.
    - **Usage:** Call to read temperature; use the output for the unit you need.



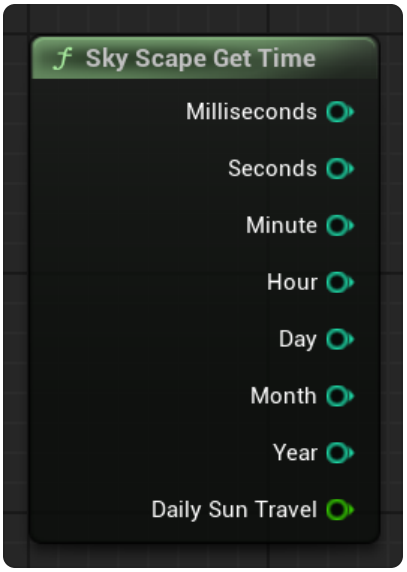
- **Outputs:**
  - **Celsius** — °C (float)
  - **Fahrenheit** — °F (float)
  - **Kelvin** — K (float)

16. **Sky Scape Get Temperature String**
    - **Description:** Returns the current air temperature as **formatted strings**, using the chosen decimal separator and including units.
    - **Usage:** Set **Separator** to `"."` or `","`; call the node to get UI-ready text.

    - **Inputs:** **Separator** — Decimal delimiter to use (`"."` or `","`).
    - **Outputs:**
      - **Celsius** — String like `"23.5 °C"`
      - **Fahrenheit** — String like `"74.3 °F"`
      - **Kelvin** — String like `"296.7 K"`
17. **Sky Scape Get Season**
    - **Description:** Returns the parameters of the currently active season as a struct.
    - **Usage:** Call to read the active season; break the struct to inspect fields.



18. **Sky Scape Get Time**
    - **Description:** Returns the current simulation time components as integers, plus **Daily Sun Travel** as the current hour in decimal form (e.g., **7:30 → 7.5**).
    - **Usage:** Call to read time for UI/logic; use **Daily Sun Travel** for sun/lighting curves or time-based drives.



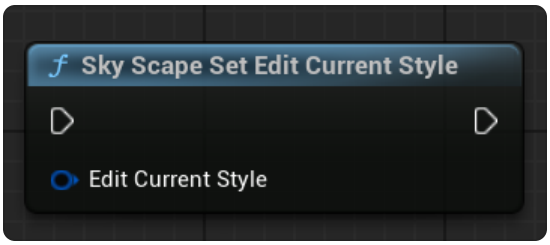
- **Outputs:**
  - **Milliseconds** — int
  - **Seconds** — int
  - **Minute** — int
  - **Hour** — int
  - **Day** — int
  - **Month** — int
  - **Year** — int
  - **Daily Sun Travel** — float (decimal hours)

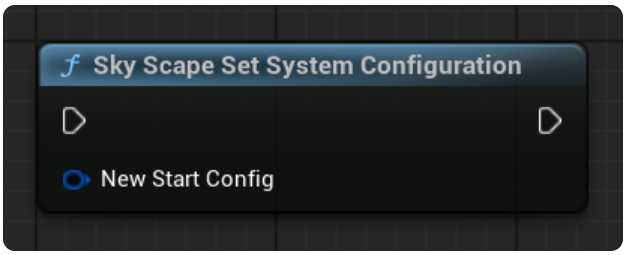
19. **Sky Scape Get Time String**
    - **Description:** Returns the current simulation time components as **strings** (no units).
    - **Usage:** Call for UI-ready text; concatenate Hour/Minute/etc. as needed.

    - **Outputs:**
      - **Milliseconds** — String
      - **Seconds** — String
      - **Minute** — String

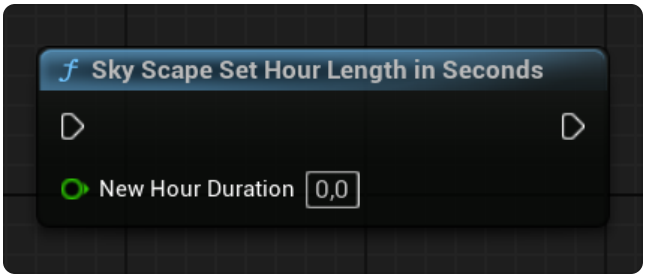
- **Hour** — String
- **Day** — String
- **Month** — String
- **Year** — String

20. **SkyScape Set Parameter**
    - **Description**: Allows you to set specific parameters for the SkyScape system.
    - **Usage**: Use this function to update or customize individual SkyScape settings dynamically.

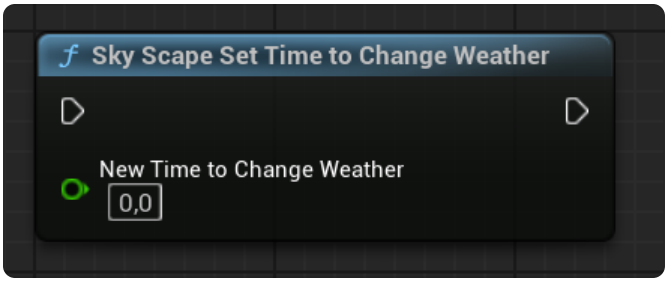




21. **SkyScape Set Hour Lenght In Seconds**
    - **Description**: Changes the **Hour Lenght In Seconds** in SkyScape, which defines how long an in-game hour lasts in real-world seconds.
    - **Usage**: Call this function to speed up or slow down the in-game time progression.

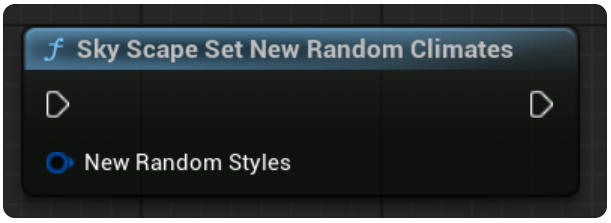


22. **SkyScape Set Time To Change Weather**
    - **Description**: Modifies the **Time To Change Weather**, which determines how long it takes for the system to transition from one weather condition to another.
    - **Usage**: Adjust this to control the smoothness or speed of weather transitions.

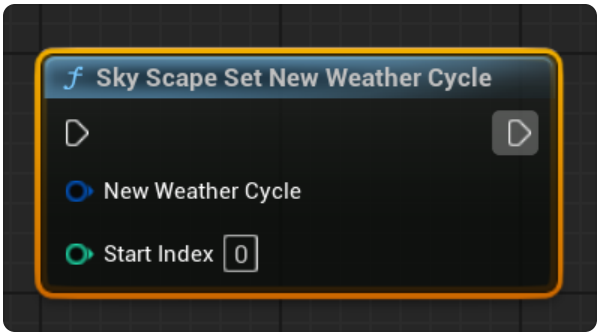


23. **Sky Scape Set Season Array**
    - **Description:** Takes an array of the Season structure, allowing you to define and change the active seasons in your project.
    - **Usage:** Provide an array of Season structures to configure how the system handles each season transition and behavior.

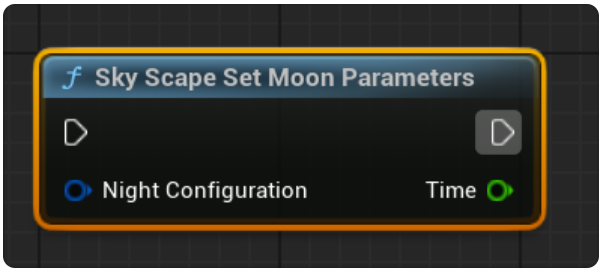
24. **Sky Scape Set New Random Climates**
    - **Description:** Allows you to change the array of random climates. Its input is an array of `editCurrentStyle` structures used to define the system's random climate.
    - **Usage:** Provide an array of `editCurrentStyle` structures to specify which random climate patterns the system can cycle through.



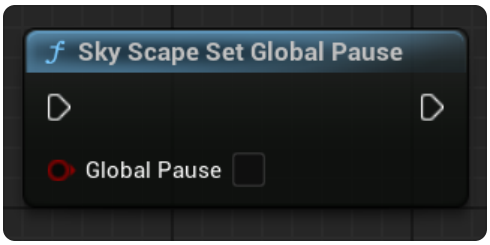
25. **Sky Scape Set New Weather Cycle**
    - **Description:** Allows you to change the plugin's sequential cycles. Its main input is an array of `editCurrentStyle` that defines the sequence, and an index to specify which item in the array to start from.
    - **Usage:** Provide an array of `editCurrentStyle` structures to define the weather cycle sequence and set the start index to pick the initial element in that sequence.



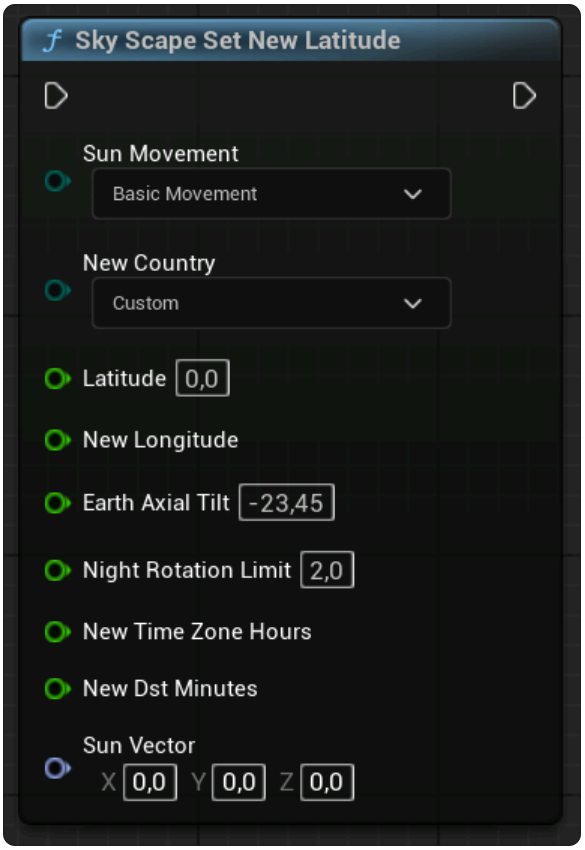
26. **Sky Scape Set Moon Parameters**
    - **Description:** Accepts an `FNightConfiguration` structure to adjust the basic parameters of the moon and stars.
    - **Usage:** Provide an `FNightConfiguration` instance to define how the moon and stars are displayed in your scene.



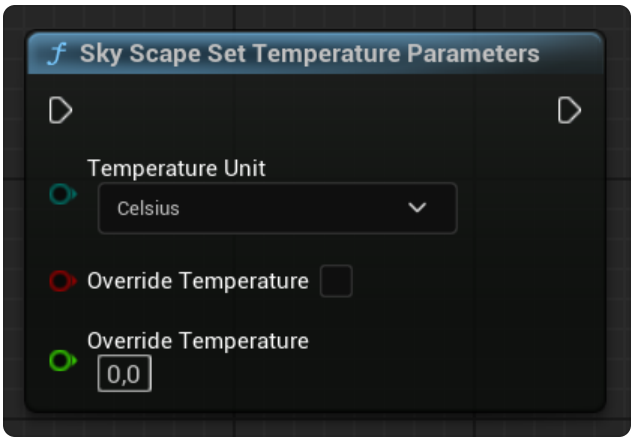
27. **SkyScape Set Global Pause**
    - **Description:** This node toggles the **Global Pause** flag in the SkyScape system. When **Global Pause** is **enabled**, **all timers and automatic weather transitions are halted**; when **disabled**, the system **resumes** its weather cycle.
    - **Usage:** Check the **Global Pause** box to **freeze** (true) or uncheck it to **resume** (false) the weather-change logic at runtime.



28. **SkyScape Set New Latitude**
    - **Description:** Configures a **new geographic latitude** and related parameters in the SkyScape system. Based on the chosen **"Sun Movement"** mode and **"New Country"** preset (or custom values), it updates the **Latitude**, **Earth Axial Tilt**, and **Night Rotation Limit**, then recalculates the **Sun Vector** to match the new location.
    - **Usage:** Use this node to **change the simulated latitude** at runtime—select a predefined country or choose "Custom" and input your own **Latitude** and **Axial Tilt** values. The system will output an updated **Sun Vector** so that sunrise, sunset, and night cycle align with the new latitude.



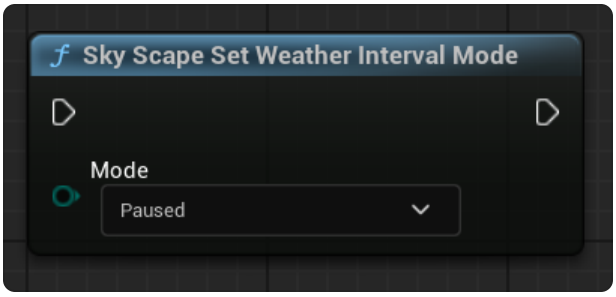
29. **SkyScape Set Temperature Parameters**
    - **Description:** Defines the **temperature unit** (Celsius, Fahrenheit, or Kelvin) used by the SkyScape system and optionally **overrides** the dynamically calculated temperature with a **fixed value**. When **"Override Temperature"** is enabled, the system will ignore its procedural temperature and use the specified **Override Temperature** instead.
    - **Usage:** Use this node to **switch** between temperature units or to **force** a specific temperature:
        i. Choose your **Temperature Unit** from the dropdown.
        ii. Tick **Override Temperature** if you want to lock in a custom value.
        iii. Enter your desired **Override Temperature** (in the chosen unit).



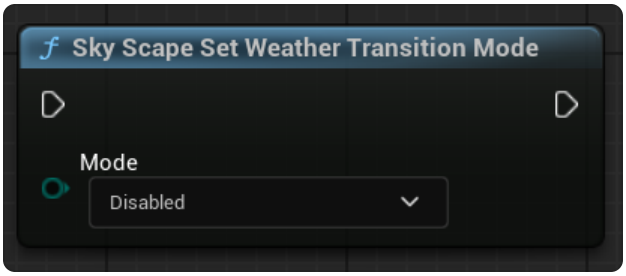
30. **SkyScape Set Weather Interval Mode**
    - **Description:** Selects how and when SkyScape will **automatically change** weather:
        - **Paused: Disables** all automatic weather changes.

- **Random Interval:** Picks a **random** delay (between the configured minimum and maximum) before each weather change.
- **Fixed Delay:** Uses a **constant interval** (the same duration every time) for weather transitions.

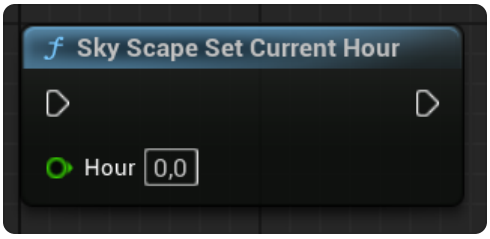
- **Usage:**
  i. Open the **Mode** dropdown.
  ii. Choose **Paused**, **Random Interval**, or **Fixed Delay** to control your weather cycle behavior.



31. **SkyScape Set Weather Transition Mode**
    - **Description:** Defines **how** SkyScape will **pick the next weather style** each time it changes:
      - **Disabled:** **No automatic** weather transitions occur.
      - **Random:** Selects a **random** style from your list of presets.
      - **Sequential:** Cycles **in order** through your presets (0 → 1 → 2 → … → 0).
      - **Weighted:** Chooses a style based on **user-assigned weights**, so some presets occur **more often** than others.
    - **Usage:**
      i. Open the **Mode** dropdown.
      ii. Pick **Disabled**, **Random**, **Sequential** or **Weighted** to control your transition strategy.



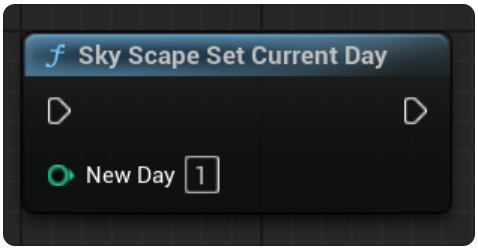
32. **Sky Scape Set Current Hour**
    - **Description:** Immediately sets the current in-game hour to the provided value.
    - **Usage:** Provide **Hour** and execute the node. Decimals are allowed (e.g., **7.5 → 07:30**).



    - **Inputs:** **Hour** — Target hour (float, decimal hours).

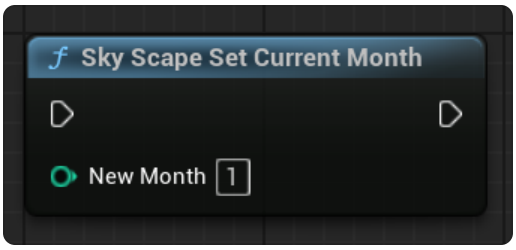
33. **Sky Scape Set Current Day**
    - **Description:** Immediately sets the current in-game day to the provided value.

- **Usage:** Provide **New Day** and execute the node.



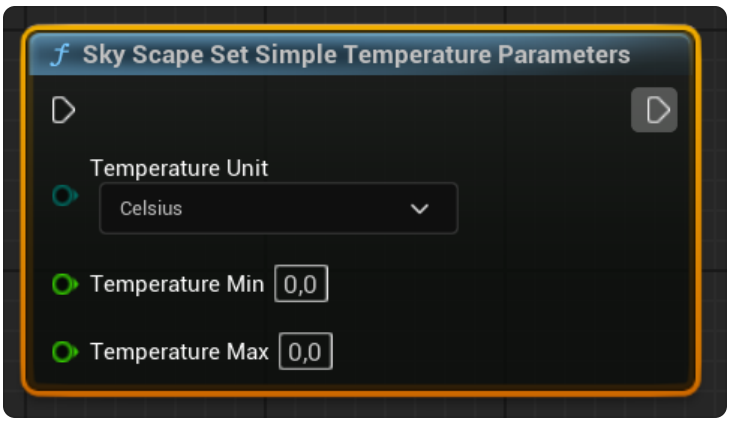
- **Inputs:** **New Day** — Target calendar day (number).

34. **Sky Scape Set Current Month**
    - **Description:** Immediately sets the current in-game month to the provided value.
    - **Usage:** Provide **New Month** and execute the node.

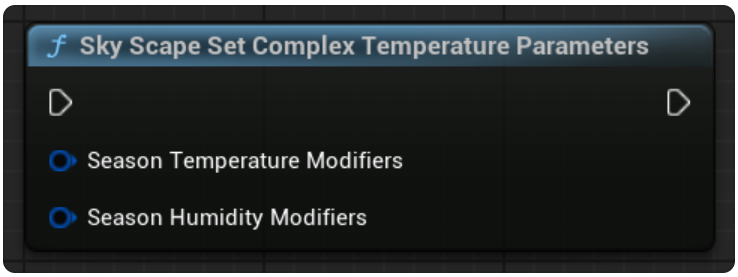


    - **Inputs:** **New Month** — Target month (integer).
35. **Sky Scape Set Current Year**
    - **Description:** Immediately sets the current in-game year to the provided value.
    - **Usage:** Provide **New Year** and execute the node.

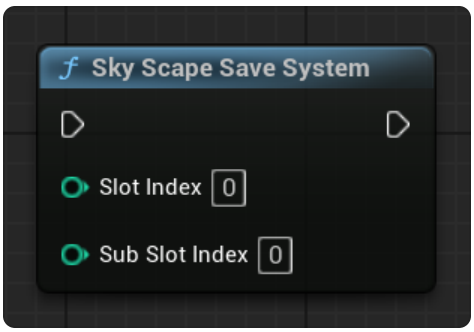
    - **Inputs:** **New Year** — Target year (integer).
36. **Sky Scape Set Simple Temperature Parameters**
    - **Description:** Configures the **Simple** temperature model: unit and min/max bounds used to generate the temperature.
    - **Usage:** Pick **Temperature Unit**, set **Temperature Min/Max** (in that unit), then execute to apply.



- **Inputs:**
  - **Temperature Unit** — Celsius / Fahrenheit / Kelvin.
  - **Temperature Min** — Lower bound (float, uses selected unit).
  - **Temperature Max** — Upper bound (float, uses selected unit).

37. **Sky Scape Set Complex Temperature Parameters**
    - **Description:** Configures the **Complex** temperature model by supplying the season-level modifiers for temperature and humidity calculations.
    - **Usage:** Build/fill the two modifier structs and plug them in; execute to apply. Effective when Temperature Mode is **Complex**.
    - 

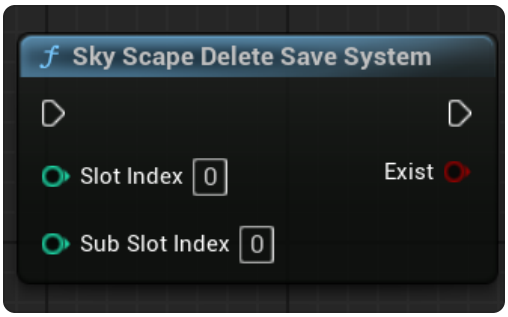
    - **Inputs:**
      - **Season Temperature Modifiers** — Struct with advanced temperature terms (e.g., solar/longwave/advection/convection/surface/latent/turbulence/anthropogenic weights, interpolation factors).
      - **Season Humidity Modifiers** — Struct with humidity dynamics (e.g., rain recharge, dew deposition, snow-melt to water, external recharge, evaporation/sublimation rates).
      - **Notes:** Ignored in **Simple** mode.
38. **SkyScape Save System**
    - **Description:** Saves the **entire SkyScape state** — including current weather style, time, and system configuration — into the specified **main slot** and **sub-slot**.
    - **Usage:**
      i. Set **Slot Index** to choose the **primary save slot**.
      ii. Set **Sub Slot Index** to choose the **secondary sub-slot** within that save.
      iii. Call the node to **persist** all relevant SkyScape data for later loading.



39. **SkyScape Delete Save System**
    - **Description:** Removes the saved SkyScape data from the specified **main slot** and **sub-slot**, and outputs whether that save **existed** (and was deleted).
    - **Usage:**
      i. Set **Slot Index** to choose the **primary save slot**.

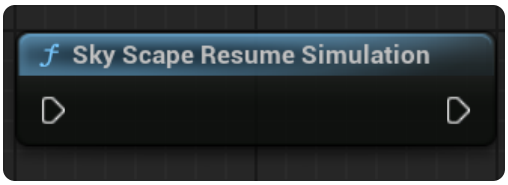
ii. Set **Sub Slot Index** to choose the **secondary sub-slot**.

iii. Call the node. The **Exist** output will be **true** if a save was found (and has now been deleted), or **false** if no save existed in that slot.



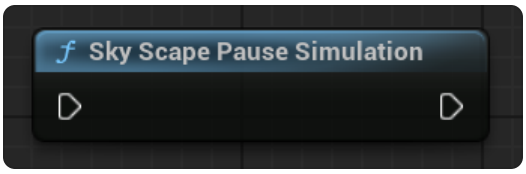
40. **SkyScape Pause Simulation**

- **Description:** Immediately **pauses** all ongoing SkyScape dynamics—**time progression**, **weather interpolation**, and **scheduled loops**—freezing the simulation in its current state.
- **Usage:**
  i. Invoke this node when you need to **halt** the SkyScape system (for cutscenes, debugging, or performance reasons).
  ii. All **timers**, **transitions**, and **effects** will stop until you resume.



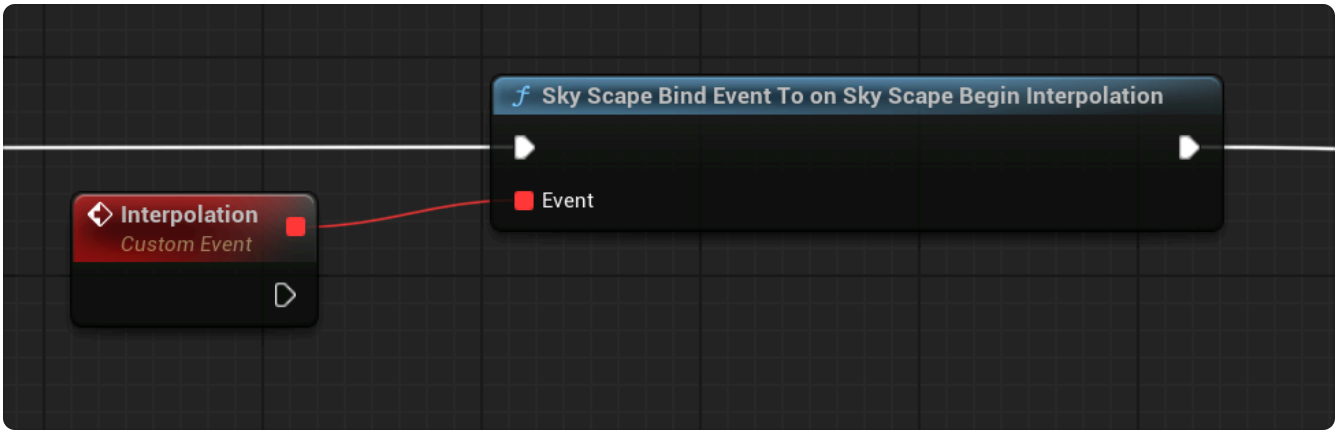
41. **SkyScape Resume Simulation**

- **Description:** **Unpauses** the SkyScape system, **restarting** all paused **timers**, **weather transitions**, and **time progression** exactly where they left off.
- **Usage:**
  i. After pausing, call this node to **continue** the simulation seamlessly.
  ii. All **interpolations** and **loops** will pick up from their previous state.



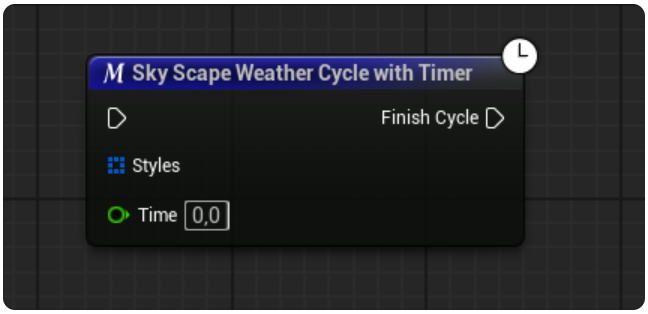
42. **SkyScape Bind Event To on SkyScape Begin Interpolation**

- **Description:** Subscribes a Custom Event to the delegate fired when SkyScape **begins** an interpolation. No parameters.
- **Usage:**
  i. Bind in `BeginPlay` of the persistent actor.
  ii. Avoid duplicate binds; duplicates cause multiple calls.

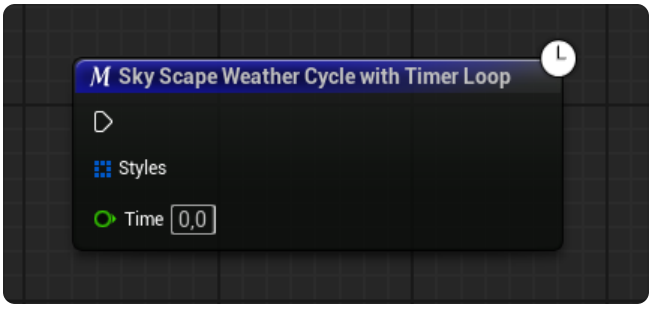


Now a series of macros to emulate behavior:

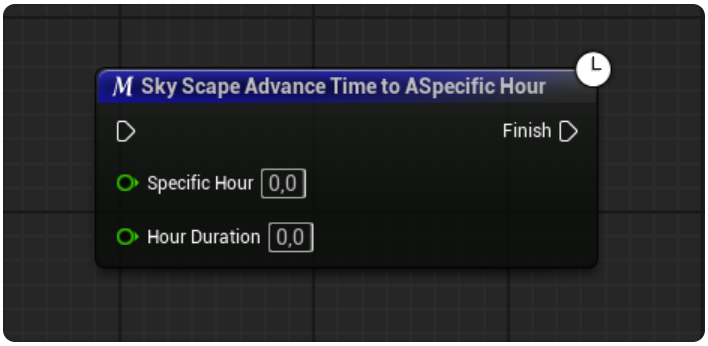
1. **SkyScape Weather Cycle With Timer**
   - **Description**: This macro allows you to create a weather cycle using an **array of styles** and a specific time interval between them.
   - **Usage**: Works similarly to the Sequential Climate system, transitioning through the provided styles in the order defined in the array.



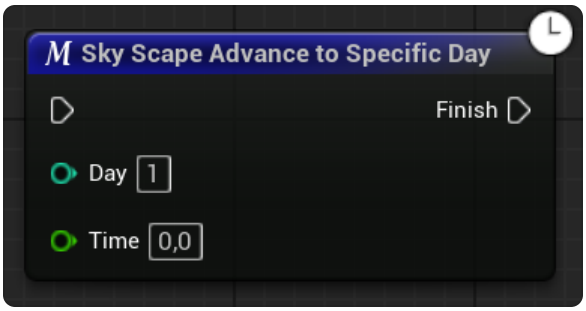
2. **SkyScape Weather Cycle With Timer Loop**
   - **Description**: Similar to the previous macro, but it **loops** the sequence. Once it reaches the last style in the array, it restarts from the first style.
   - **Usage**: Ideal for creating a continuous and repeating sequence of weather changes.



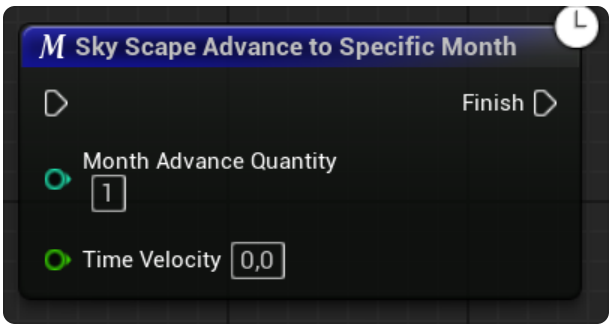
3. **SkyScape Advance Time To A Specific Hour**
   - **Description**: Advances the in-game time to a specific hour. You define the **target hour** and the **speed** at which the time should progress until it reaches the desired hour.
   - **Usage**: Useful for skipping to a specific time of day dynamically.



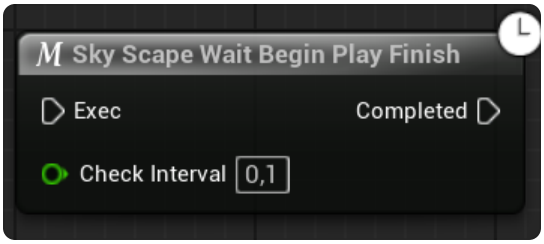
4. **Sky Scape Advance to Specific Day**
   - **Description**: **Advances** the in-game simulation calendar to a **specific day** and **time**, then fires the **Finish** output once reached.
   - ***Usage***: Connect the desired **Day** and **Time** values; use the **Finish** pin to continue your Blueprint flow when that date/time is achieved.



5. **Sky Scape Advance to Specific Month**
   - **Description:** **Advances** the in-game simulation calendar by a **specified number of months** at a given speed, then fires the **Finish** output once the month increment is complete.
   - **Usage:** Set **Month Advance Quantity** to how many months to jump, and **Time Velocity** to control the speed; hook up **Finish** to proceed when done.



6. **Sky Scape Wait Begin Play Finish**
   - **Description:** This macro **pauses** execution until the Sky Scape actor has fully completed its **BeginPlay** initialization. Internally it **polls** every **Check Interval** seconds and then fires the **Completed** output once initialization is done.
   - **Usage:**
     i. Plug your execution flow into **Exec**.
     ii. Set **Check Interval** to control how often it checks (e.g. 0.1 s).
     iii. Connect logic to the **Completed** pin — it will only run after Sky Scape's BeginPlay has finished.



Control de SkyScape In-Game: Cambios de Estilo y Ajustes desde el Personaje

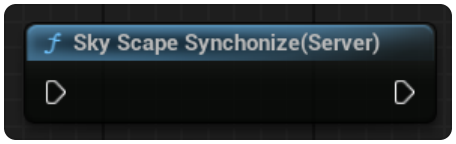
Multiplayer

In the online version of our UE5 weather plugin, every function now comes with its dedicated **server counterpart**, and these functions must always be called from the server. This ensures that all of the capabilities you see—from changing the weather to adjusting in-game time—are fully supported in multiplayer environments. For example, functions like **SkyScape Change Weather** and **SkyScape Change Weather And Get Time** have their respective server versions, ensuring that any weather changes or transitions triggered on one client are consistently reflected across the entire network.

Each core function, such as **SkyScape Get Reference**, **SkyScape Get Current Parameters**, and **SkyScape Set Parameter**, is paired with an equivalent on the server. This dual implementation provides robust control over environmental parameters, allowing for real-time updates that are synchronized between the client and the server. Additionally, time-related functions like **SkyScape Set Hour Lenght In Seconds** and **SkyScape Set Time To Change Weather** now feature server-specific versions, which are critical for maintaining precise time progression and smooth transitions in an online setting.

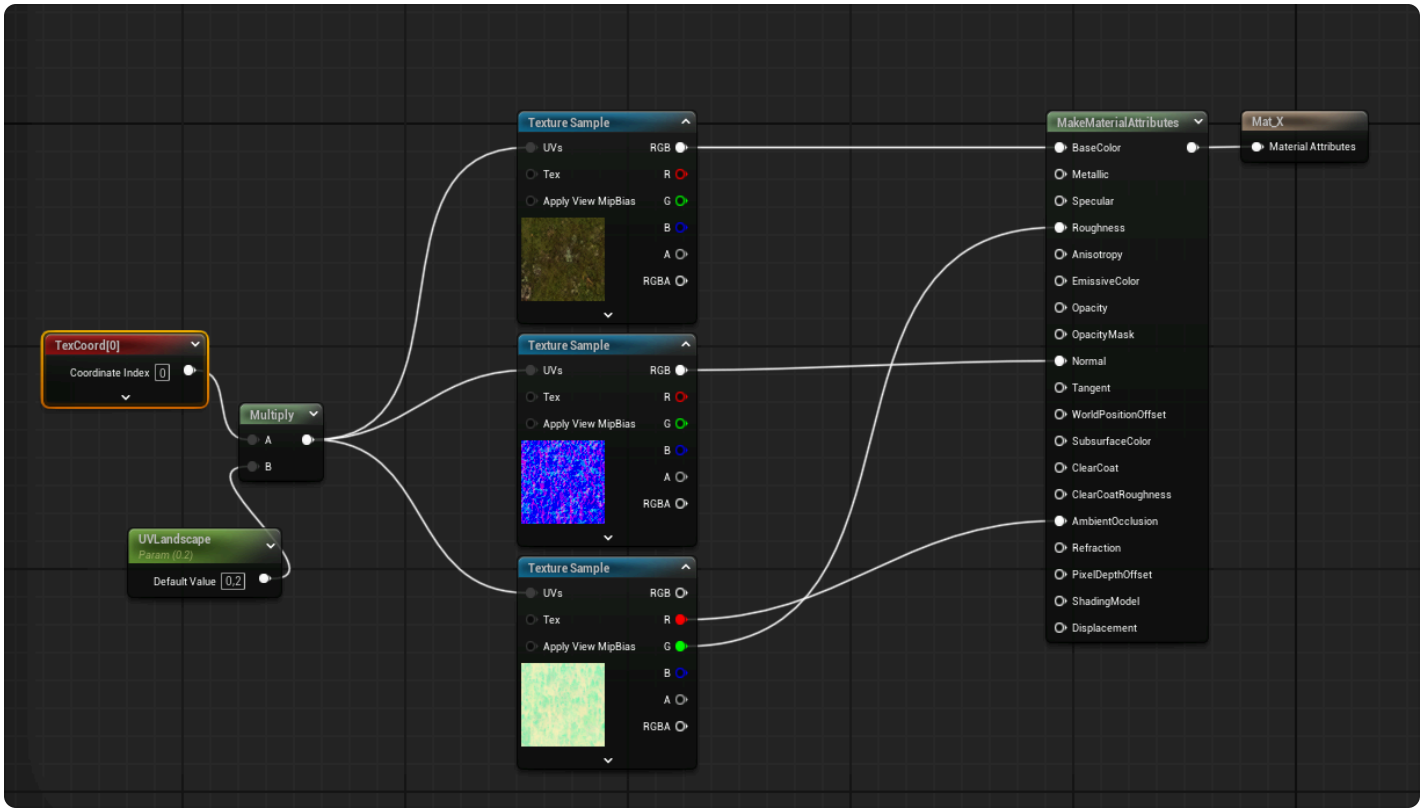
Overall, this comprehensive integration means that every function you rely on for local gameplay now has a robust **server equivalent**, ensuring seamless and synchronized weather and time management in both single-player and multiplayer experiences.

Additionally, a new function called **SkyScape Synchronize** has been added. This function synchronizes all users to the weather conditions seen on the server. It ensures that all players experience the same weather in real time, preventing desynchronization and improving consistency in the multiplayer environment.

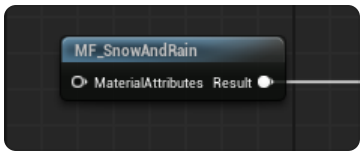


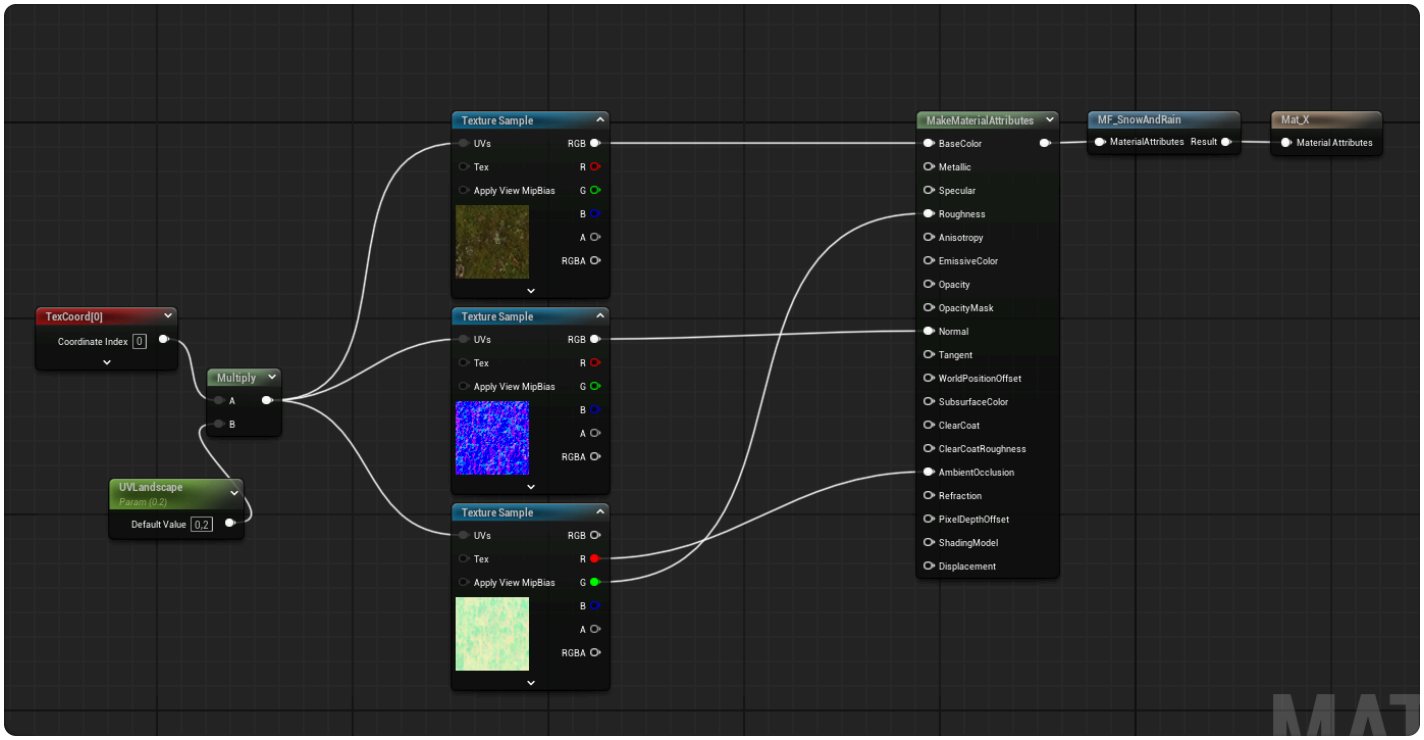
Set Landscape Materials for Snow and Rain

To set the **landscape material**, we first need an **already created material**.

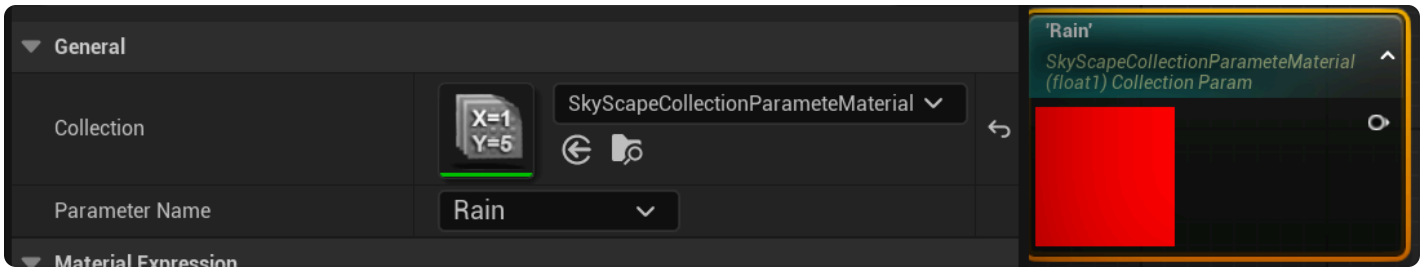


Now, we need to connect the **material function** from **MF_SnowAndRain**.





And with this, it would be ready. Additionally, in the **SkyScape dependencies folder**, there is a **Material Parameter Collection** which includes **Rain** and **Snow** as parameters, in case you, as a developer, decide to create your own rain and snow material.



SkyScape: Materiales Dinámicos según el Clima y Estaciones

Package SkyScape

To package **SkyScape**, we need to place a structure called **NecessaryObjects** as a variable in our level. This ensures that **hard references** are created in the project for the objects contained in this structure, guaranteeing that they are included in the **packaging process**.

SkyScape: Packaging & Publicación

Attributions

- **8K Sun Texture**
  – **Source:** Solar System Scope
  – **License:** Creative Commons Attribution 4.0 International (CC BY 4.0)
- https://creativecommons.org/licenses/by/4.0/
  – **Download:** https://www.solarsystemscope.com/textures/
- **8K Moon Texture**
  – **Source:** Solar System Scope
- – **License:** Creative Commons Attribution 4.0 International (CC BY 4.0)
- https://creativecommons.org/licenses/by/4.0/
  – **Download:** https://www.solarsystemscope.com/textures/